A STUDY OF THE SYSTEM DEVELOPMENT PROCESS

Gabriel Oswaldo Flores-Prado

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A Study of the System Development Process

by

Gabriel Oswaldo Flores Prado

December 1977

Advisor: N. F. Schneidewind

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A Study of the System Development Process		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis; December 1977
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Gabriel Oswaldo Flores Prado		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		12. REPORT DATE December 1977
		13. NUMBER OF PAGES 167
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Management information is of critical importance in modern decision making. The role of the computer in this process is rapidly expanding, creating challenging goals for data processing specialists and functional area specialists alike. A totally integrated computer based management information system (MIS) requires long term planning and design efforts coupled

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |
(Page 1)

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

with detailed analysis of information system requirements. The MIS development generally follows a master plan; this master plan contains three major phases: MIS Analysis, MIS Design, and MIS Implementation. The different phases through which the master plan evolves are known as the system development process. This thesis describes the development of the master plan.

Approved for public release; distribution unlimited.

A Study of the System Development Process

by

Gabriel Oswaldo Flores-Prado
Lieutenant Commander, Venezuelan Navy
B.S., Naval Postgraduate School, 1977

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the
NAVAL POSTGRADUATE SCHOOL
December, 1977

ABSTRACT

Management information is of critical importance in modern decision making. The role of the computer in this process is rapidly expanding, creating challenging goals for data processing specialists and functional area specialists alike. A totally integrated computer based management information system (MIS) requires long term planning and design efforts coupled with detailed analysis of information system requirements. The MIS development generally follows a master plan; this master plan contains three major phases: MIS Analysis, MIS Design, and MIS Implementation. The different phases through which the master plan evolves are known as the system development process. This thesis describes the development of the master plan.

TABLE OF CONTENTS

I. INTRODUCTION

In the middle 1950's the first computer was installed for a business application: processing of payroll. Twenty years later there were over 100,000 computers in the United States and a like number in the rest of the world. Payroll processing by computer, which was a revolutionary idea in 1954, is now considered a rather routine application. Today the frontiers in information processing are systems which also provide information resources in support of managerial and decision-making functions. Such a system is called a Management Information System or, more commonly, MIS.

The key to managerial success in the present automated environment lies in being able to collect, quantify, retrieve and effectively use the information available in the decision making process.

A. OBJECTIVES

This discourse is an attempt at a sketch of that body of knowledge known as system analysis. The kind of system analysis that is examined here is that which deals with information systems.

A discussion of the stages of building a management information system (referred to hereafter as MIS) should provide a reference or guide for those who are presently involved in future information system development.

A step-by-step discussion of the System Life Cycle will be presented and an attempt will be made to utilize it in the development of a MIS.

Systems Integration will be the point which will be stressed throughout this discussion.

B. SYSTEM LIFE CYCLE

1. Definition of Systems

A system [1] can be defined as a network of interrelated procedures that are joined together to perform some activity, function or operation. It is, in effect, all the ingredients which make up the whole. And a procedure is a precise series of step-by-step instructions that explain:

- What is to be done.
- Who will do it.
- When will it be done.
- How will it be done.

2. Definition of Systems Analysis

System Analysis [2] is the process of studying the network of interactions within an organization and assisting in the development of new and improved methods for performing necessary work.

3. The Systems Development Process

The System Development Process is a guideline for the development of computerized systems. It is a comprehensive process beginning with the definition of a problem and including the design and implementation of a new system which will

correct the problem. It discusses the complete life cycle of a system, from its initial conception to its ultimate disposal. (See Figure 1.1.) Figure 1.1 has been derived from references [3] and [4].

The System Life Cycle consists of the following activities:

a. Conception

System Conception is the identification of a need which, it is suspected, can be satisfied by the development of a new system. At this stage the need for a new information system is first recognized.

b. Preliminary Analysis

The Preliminary Analysis is the investigation into the feasibility, desirability, and practicability of using automated data processing techniques to solve information needs.

c. Preliminary Design

The Preliminary Design is the stage in which the project team evaluates alternate design approaches, selects preferred design approach, writes preliminary design specifications and writes the test specification. The basic system is established at this stage.

d. Detailed Design

The Detailed Design is the stage where the basic system designed is expanded and refined to produce detailed specifications for all program modules, manual procedures, and information files.

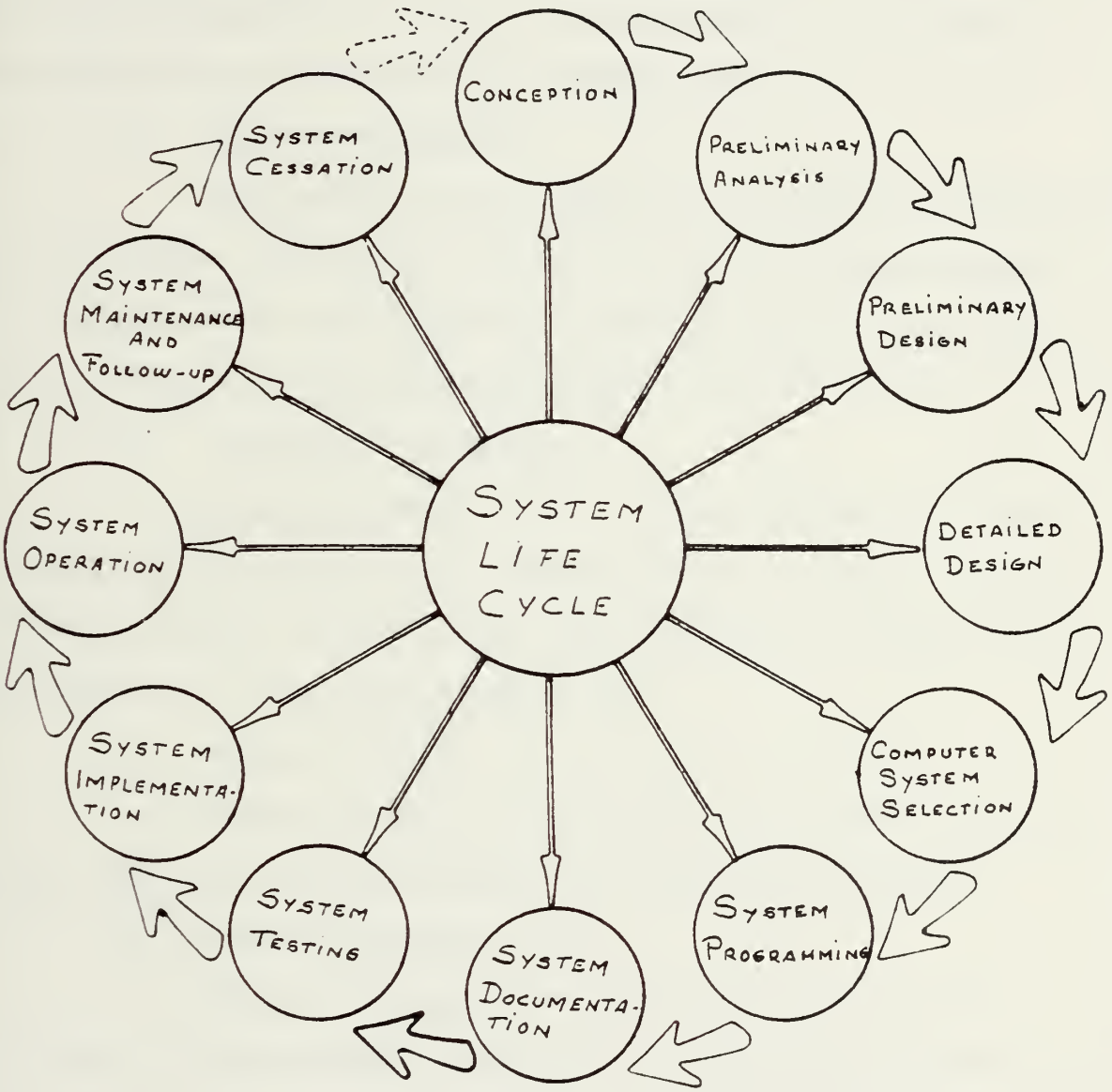


FIG. 1.1

SOURCE: REFERENCES [3], [4]

e. Configuring and Selecting the Computer Equipment

The previous stages were the basis for setting up exactly which applications are to run on the system.

At this stage, the configuration of the computer System (Hardware/Software) is established and the selection and evaluation of vendors are accomplished.

f. System Programming

The logic for the computer's programs is developed at this stage, following the system specifications accepted by the user. Once the logic has been set up the programmers begin coding and testing each program in the system.

g. System Documentation

System Documentation is the collection and presentation of the information concerning the system. Information is recorded starting from the conception of the system and ending with the cessation of it.

h. System Testing

System Testing is the process of trying to demonstrate that the system performs its intended function.

i. System Implementation

System Implementation consists of the tasks involved in implementing the system in the operating environment.

j. System Operation

System Operation is the production phase. Responsibility for the system is now shifted to the operations

group. However, the designers and programmers can contribute to making the transition easy.

k. System Maintenance and Follow-up

Once the system has been made operational, it is important to have a continuing support service to maintain the hardware and software.

1. System Cessation

System Cessation is the end of the System Life Cycle. The system approach assumes that all systems have a finite life and, therefore, will eventually expire.

The application of the System Life Cycle to the development of a MIS will be discussed later.

II. MANAGEMENT INFORMATION SYSTEMS

A. DEFINITION OF DATA

Data are isolated facts which could represent quantities, actions, things, etc., but which are in unusable form for the recipient unless they are submitted to some sort of processing.

B. DEFINITION OF INFORMATION

Information [5] is data that has been processed into a form that is meaningful to the recipient and is of real or perceived value in current or perspective decisions.

C. PRODUCING INFORMATION FROM DATA

It has been established that data need to be processed in order to transform it in meaningful information for the recipient. This transformation can be viewed from two different viewpoints. First a logical viewpoint, the operations performed on data and second, a technical viewpoint, the methods by which data operations are performed.

1. Data Operations

In order to make the data useful to the recipient, some combinations of operations need to be performed on it. The following are the set of operations, as described by Burch [6]:

a. Capturing

Operation of recording the data from an event or occurrence in some form such as sales slips, personnel forms, purchase orders and so forth.

b. Verifying

Operation for checking that data have been recorded correctly.

c. Classifying

Operation that places data elements into specific categories which provide meaning for the user.

d. Sorting

Operation that arranges data elements in a specified or predetermined sequence.

e. Summarizing

Summarizing can be done in two ways: First, it accumulates data in the mathematical sense, as when a balance sheet is prepared. Second, it reduces data in the logical sense, as when the personnel manager wants a list of names of only the employees who are assigned to a given department.

f. Calculating

Operation that deals with the arithmetic and/or logical manipulation of data.

g. Storing

Operation that places data onto some storage media such as paper, microfilm, and magnetizable devices, where it can be kept for access and retrieval when needed.

h. Retrieving

This operation entails searching out and gaining access to specific data elements from the medium where they are stored.

i. Reproducing

This operation duplicates data from one medium to another, or into another position in the same medium. An example of this operation is the duplication of files for security reasons.

j. Disseminating/Communicating

This operation transfers data from one place to another. The ultimate aim of all data processing is to disseminate information to the users who need it.

2. Data Processing Methods

In order to carry out the data operations previously described, four broad categories, based on the level of automation used for processing the data, can be defined:

a. Manual Method

All of the data operations are performed by hand with the aid of basic devices such as pencil, paper, slide rule, and so forth.

b. Electromechanical Method

It is actually a symbiosis of man and machine. Example of this method includes an operator working at a cash register.

c. Punched Card Equipment Method

It entails the use of all devices used in what is sometimes referred to as a unit record system. Data concerning a person, object, or event are normally recorded (punched) in a card.

d. Electronic Computer

All of the data operations, are performed by an electronic computer. The computer provides significantly greater data processing capabilities that the other three methods.

A table showing the relationship between the data operation and the data processing methods can be seen in Figure 2.1.

D. VALUE OF INFORMATION

Information is a valuable resource in any organization. Without formal information most organizations could not survive.

Decisions can be made under certainty, risk and uncertainty. Decision under certainty assumes perfect information is available concerning outcomes; risk assumes information is available about the probability of each outcome but no knowledge of possible outcomes; and uncertainty assumes a knowledge of possible outcomes but no information about probability of outcome [5].

Information can be studied in terms of its cost and value. While the cost of providing information is tangible

Operations \ Methods	Manual Method	Electro-mechanical Method	Punched Card Equipment Method	Electronic Computer Method
Capturing and Initial Recording	Voice; observation; handwritten records; forms and checklists; writing boards; pegboards	Typewriter; cash register; autographic registers; time clocks	Key punch; verifier; mark-sensed cards; prepunched cards; machine readable tags	Key punch; verifier; paper tape punch; magnetic encoder; OCR enscriber; collection devices; conversion devices; terminals
Classifying	Hand posting; coding; identifying pegboards	Posting machine; cash register; accounting machine	Sorter; collator	By systems design
Arranging	Alphabetizing; indexing; filing; edge notched cards	Semi-automatic (roto-matics; gather-matics)	Sorter; collator	Card sorter; internal computer sorting
Summarizing	Hand calculators	Adding machines; calculators; cash registers; posting machines	Accounting machine; calculator summary punch	Central processing unit
Calculating	Human calculation; pencil and paper; abacus; slide rule	Accounting machines; adding machines; calculators; cash registers; posting machines	Accounting machine; calculator summary punch	Central processing unit

Figure 2.1 Data Processing Methods
Source: ref [6]

Operations \ Methods	Manual Method	Electro-mechanical Method	Punched Card Equipment Method	Electronic Computer Method
Storing	Columnar journals ledgers; index cards; paper files	Mechanical files (rotary or tub files); microfilm	Card trays	CPU DASD; magnetic tape; paper tape; punched cards
Retrieving	File clerks; stock clerks; book keepers	Methcnical files (rotary or tub files); microfilm	Sorter; collator; hand selection	Online inquiry into DASD; report generation
Reproducing	Hand-copying; carbon paper	Duplicating equipment (carbonization, hectograph, stencil, offset, photocopying thermograph); addressing equipment	Reproducers; interpreter	Same as before, plus on line copies from line printer; computer input/output; microfilm
Disseminating and Communicating	Hand-written reports; hand-carried, or mailed	Telephone; teletype; machine prepared reports; message conveyors; hand-carried or mailed reports	Same as before	Same as before plus on line data transmission (telecommunication); visual display; voice output

Figure 2.1 (Continuation)

and fairly measurable, its value is conceptual in nature and has less tangible characteristics.

1. Cost of Information

Information is not free. The preparation of information for decision making costs money. The following are some of the elements that need to be considered when estimating the cost of producing information, for a computer-based MIS:

- Cost of the computer system.
- Cost of developing the system.
- Cost for on-site preparation.
- Cost of conversion from the present system to a computer-based system.
- Cost of operation.

2. Value of Information

The value of information is based on its attributes, and because of the conceptual nature of information some of those attributes are difficult to measure. The list of these attributes, as given by Burch [6] follow:

a. Accessibility

This refers to the ease and speed with which an information output can be obtained.

b. Comprehensiveness

This refers to the completeness of the information content. This attribute is quite intangible and consequently, it is difficult to quantify.

c. Accuracy

This attribute pertains to the degree of freedom from error of the information output.

d. Appropriateness

This attribute refers to how well the information output relates to the user's request.

e. Timeliness

The user must receive the information, within the time allowed, for the decision or action to which it is applied.

f. Clarity

This attribute refers to the degree an information output is free from ambiguous terms.

g. Flexibility

This attribute pertains to the adaptability of an information output not only to more than one decision, but to more than one decision maker.

h. Verifiability

This attribute refers to the ability of several users examining an information output and arriving at the same conclusion.

i. Freedom from Bias

This attribute pertains to the absence of intent to alter or modify information in order to produce a preconceived conclusion.

j. Quantifiable

This attribute refers to the nature of information produced from a formal information system.

E. DEFINITION OF A MIS

Actually there is not universal agreement among managers and computers scientists, about what is a MIS. Some of the definitions that were found follow:

> "An MIS System, fundamentally, is a financial control system which is the principal basis of a good planning system [7]."

> "A MIS is a system of people, equipment, procedures, documents, and communications that collects, validates, operates on, transforms, stores, retrieves, and presents data for use in planning, budgeting, accounting, controlling, and other management processes for various management purposes [8]."

> "A management information system may be defined as an organized method of providing management with information needed for decisions, when it is needed and in a form which aids understanding and stimulates action [9]."

> "A MIS is an integrated, man/machine system for providing information to support the operations, management, and decision making functions in an organization. The system utilizes computer hardware and software, manual procedures, management and decision models, and a database [5]."

It can be said that each manager or author has its own definition about a MIS. Definitions abound but there is no concensus about what really constitutes a MIS. Trying to look for a definition or trying to essay a new one, would contribute to the confusion that already exists; for this reason no attempt was made to define a MIS.

F. TYPES OF INFORMATION SYSTEMS

There is no specific system envisioned when the general term of MIS is mentioned. Today many people have their own definitions. Following are brief descriptions of four basic types, as seen by Cicio [10]:

1. Formal/Automated

This type of information system is always the most sophisticated, where the machine is more the worker and the man more the designer, director, observer, or user. Electronic computers with all of the associated hardware to complement and connect them are involved in a full blown automatic data processing system.

2. Formal/Non-automated

This system includes all those conventions, procedures, media, that are formally prescribed, legally required, or by habit adhered to; but which are implemented either manually or with minor equipment. The predominant effort is clerical.

3. Informal/Formal

This system could well be a more powerful informational system in the key management sphere than any other output of any part of the entire MIS. This is the product of organized advisory staff work; the formal staff meeting where problems are discussed; and the work of formally chartered groups such as a board of directors.

4. Informal/Informal

A system of this sort can be information passed at a cocktail party, a network of personal letters, the grapevine or any other system of information exchange. This is included because to some extent it does carry information and influences management decisions.

G. STRUCTURE OF A MIS

During the past several years, the structure of a MIS has been related to the pyramidal form of the organization. Figure 2.2 shows this structure. This figure has been adapted from reference [11].

This basic structural concept reflects two ways of the MIS structure:

1. On the Basis of Management Activity

This concept deals with the three most widely used levels of management planning and controlling activities: strategic, tactical and operational.

2. On the Basis of Organizational Function

This concept deals with the functions performed in an organization. Of course this approach will depend on the particular organization under study.

For purposes of illustration the following functions are listed for a business organization:

- Marketing
- Production
- Logistics
- Personnel
- Finance
- Research and Development
- Information Processing

Whereas the operational function view of a MIS is to group all activities and information processing in an organization by function, the management activity approach is to

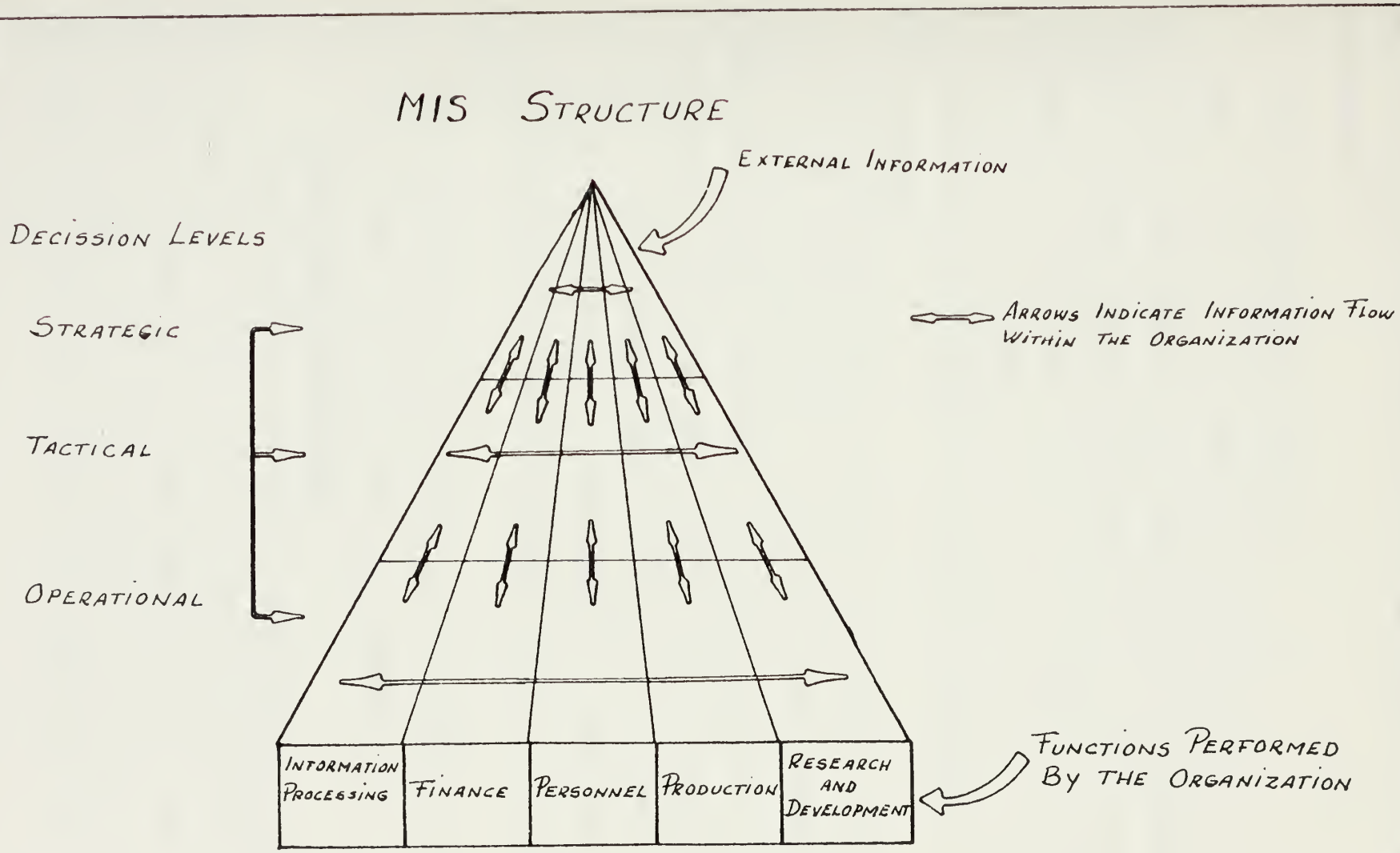


FIGURE 2.2

SOURCE: REFERENCE [11]

group activities and information system support by level of management.

H. THE DATA MANIPULATION FUNCTION IN A MIS

Data is the heart of a MIS, its establishment is essential in the development of any sophisticated information system. Data has value to the extent that it can be retrieved, processed, and presented to the person needing it within the time allowed for the decision or action to which it applies, therefore, data that cannot be located or processed on time have no value.

A MIS envisions the availability of a fairly comprehensive set of stored data in order to provide information to support operations, management, and decision making in an organization. The system must then be able to selectively retrieve various combinations of stored data on demand of the manager. The manager's requirements for information are rarely fixed and vary with the dynamic nature of the organization.

File processing systems and data base processing systems are the two basic approaches for data organization in a MIS.

1. File Processing Approach

File-processing systems are predecessors of data base processing systems. In this approach each application is processed separately by using separate files. Each data file is designed with its own storage area and also designed to provide for the needs of the users requesting the

application. Typically, the computerized files maintained by most organizations exhibit the following deficiencies [12]:

a. Lack of Integration

Data is conveniently organized into records and files for maintenance and processing, but such organization frequently fails to give recognition to its mutual relationship. Data redundancy is one of the problems due to lack of integration.

b. Lack of Program Independence

System analysts and programmers have virtually complete responsibility for designing and implementing an application. This includes, in addition to the logical design of the application, the design of all the files and records associated with it and the preparation of programs for the input, output, and updating of information. The tendency has been to design those programs in order to handle the specific data contained in the application file. The practical effect of this is that any change in data organization will require a change somewhere - usually in several places in one or more programs.

c. Unsuitability for On-line Usage

The approach to file organization in the past has been to structure sequential files consisting of physically contiguous records and sequenced according to a key. There is a serious problem in adapting existing files to on-line usage. The time required to search a sequential file varies proportionately with the size of that file, so that

for an application of any magnitude, it is impossible to provide an immediate response to a request for information. A schematic diagram, showing the file processing approach can be seen in Figure 2.3. This figure has been adapted from reference [13].

2. Database Approach

The term database is not a new one; it became current in the late 1960s. Prior to that time, the terms files of data and data sets were used. Many organizations adopted the new term to name their files of data but no changes were made in order to improve the previous established deficiencies in file processing.

Martin [14] defines a database as follows:

> "A database may be defined as a collection of interrelated data stored together without harmful or unnecessary redundancy to serve one or more applications in an optimal fashion; the data are stored so that they are independent of programs which use the data; a common and controlled approach is use in adding new data and in modifying and retrieving existing data within the database. One system is said to contain a collection of data bases if they are entirely separate in structure."

The use of a database system in a MIS will allow a centralized control of the data within the organization, data redundancy reduction, minimization of inconsistencies in the stored data, utilization of data for one or more applications, and a better logical data organization for both batch processing and on-line applications.

In addition to data organization, some other basic concepts need to be implemented in order to make the database operational.

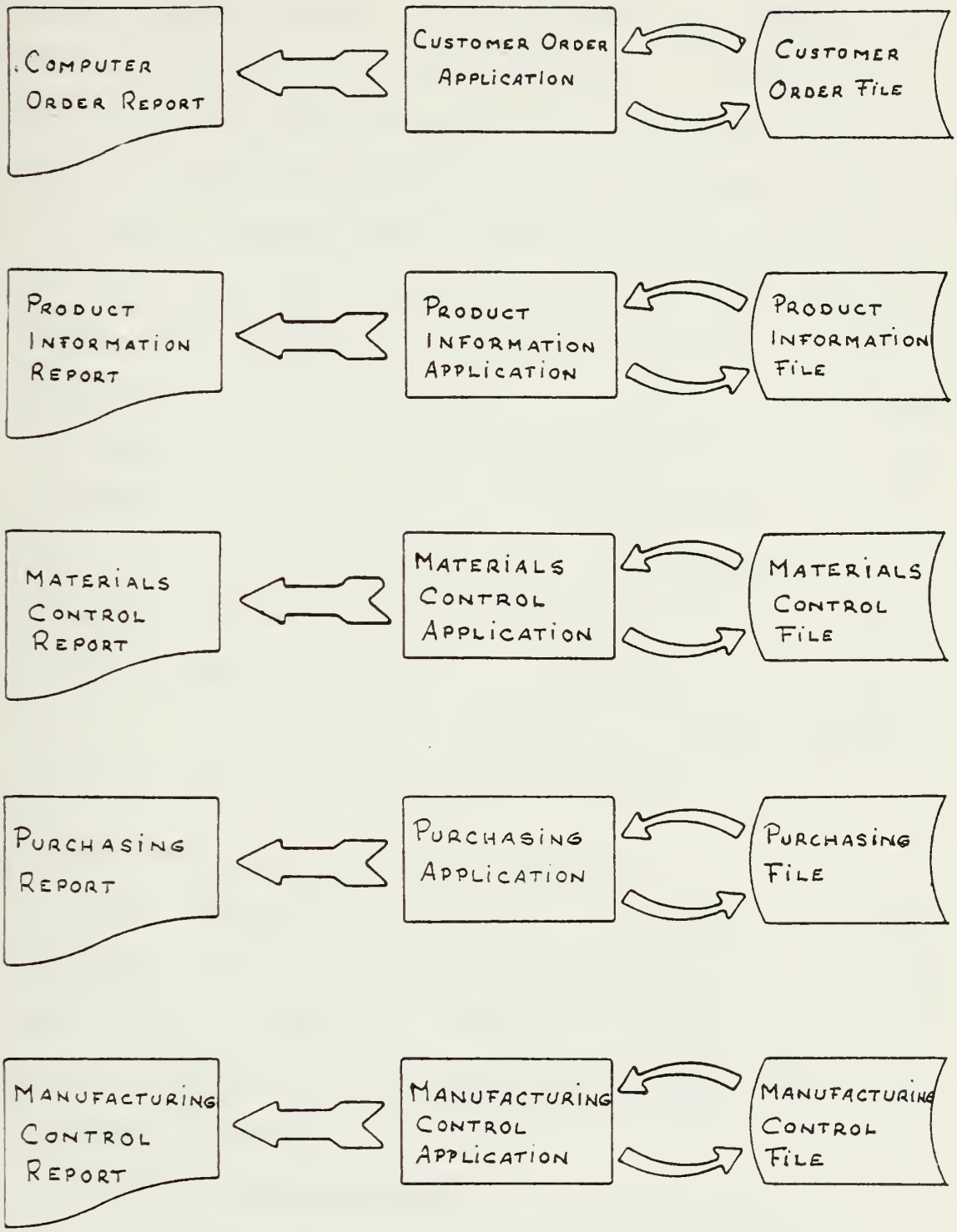


FIG. 2.3 FILE PROCESSING APPROACH

SOURCE: REF. [13]

Data organization alone is not enough for establishing a database system. It is necessary to consider the software which performs the functions of creating and updating files, retrieving data, and generating reports, which is called the database management system. It is also necessary to consider the activity of data control, which is a human function performed for the Data Base Administrator (DBA). It is important to point out that the introduction of the DBA function could lead to an organizational change. Problems could arise in the organization relative to the management level where the DBA must be placed. If this function has not been established in the organization, prime consideration must be given to this matter if the database system approach is going to be used in the MIS development. Figure 2.4 shows a simple diagram of the database approach. Figure 2.4 has been adapted from reference [13].

I. MIS OPERATING MODE CLASSIFICATIONS

Input, processing, and output are common to all computer based MIS. Timing of these actions defines the operating modes of the system [15]. Basically two modes of operation will be considered; batch processing systems and real-time processing systems.

1. Batch Processing Mode

In this mode of operation, the data that is input to the system is held for later processing rather than being processed as it becomes available; however, some kinds of

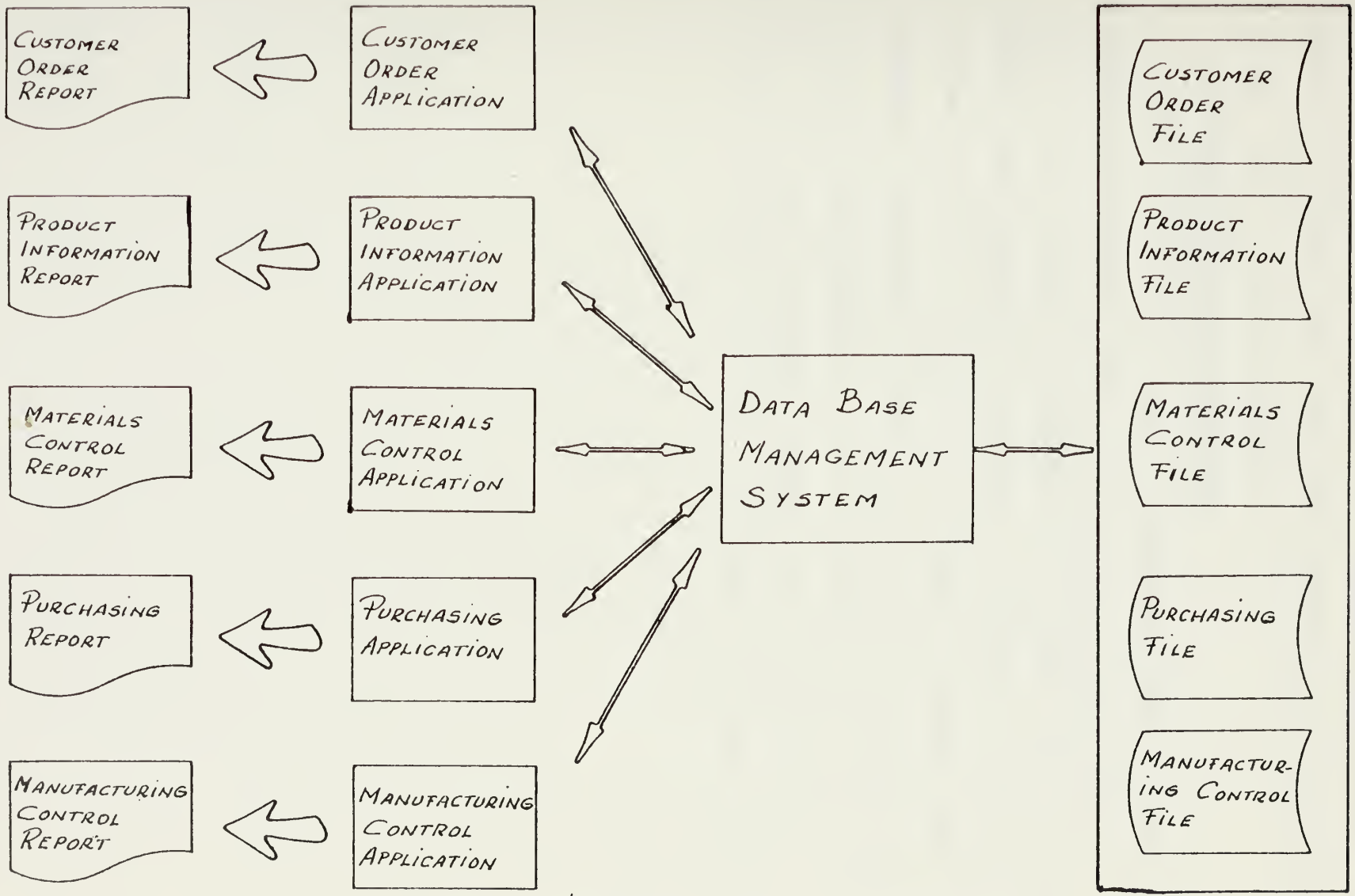


FIG. 2. DATA BASE PROCESSING APPROACH

SOURCE: REF. [13]

outputs are always produced. This happens automatically and usually at regular time intervals, depending on the particular application being processed.

2. Real-Time Processing Mode

In these systems, fast response and fast reaction of the data processing system is the chief consideration. Usually this reaction is measured in seconds, and it is the elapsed time between the time when an inquiry is made and the time in which a response is obtained. Normally these systems are expensive. Additional hardware, software, and application programming cost are associated with fast response. Of course in these systems data is processed as soon as it becomes available to the data processing system.

III. THE SYSTEM DEVELOPMENT PROCESS

It was established that the system development process is a guideline used for controlling the analysis, design and implementation of information systems within an organization. This guideline has been set up around the System Life Cycle, briefly explained in the introduction of this study.

In this section a thorough discussion of the system life cycle is presented with emphasis in its application to the MIS development process.

Before going further in this study, a brief discussion will be presented about some topics, which are of primary importance for the success of the information system. The points that will be considered are the following: management involvement, and the steering committee.

In order to be successful in the study and implementation of a MIS, both, support and participation of top management are required. Top management must support the system effort by itself, and demand support from line management and operations management if necessary. In addition, it is essential that management at all levels, participate in the different stages of the system development process.

Information analysts and managers are responsible for problems that arise when this lack of support exists. On the one hand, management is sometimes hesitant to face the system

development effort because of its lack of knowledge of computers and information systems; on the other, information analysts fail, when they do not define to the management its scope of participation, i.e., what information analysts will expect from management at the different stages of development.

A listing of the duties and responsibilities of managers and the system analysts for the various stages of MIS development will be given, when appropriate throughout this discussion, as a guide for bridging the communication gap between the system analyst and the manager.

The development of a MIS may lead to organizational problems. Some typical problems are control of expenses, determination of priorities and interdepartmental conflicts. The establishment of a steering committee has been considered one of the best ways for dealing with these problems.

The steering committee is responsible for overall control of the continuing operation of the information system function. The composition of this committee should be based on the structure of a MIS, as it was explained in the previous chapter. For each function performed within the organization, there is a top manager associated with it. These managers should be the members of the committee, and the information processing manager should act as the secretary. The chairman of the committee should be the president of the organization, or the executive at the highest level.

Functions of the permanent steering committee include the following [16]:

- It determines systems that will be automated.
- It establishes and continually reviews system development priorities.
- It reviews and approves data processing plans.
- It reviews and approves data processing equipment acquisition.
- It reviews project status.
- It decides, when necessary, if projects should be continued or abandoned.
- It carries out resources allocation policies.
- It resolves territorial and political conflicts within the organization which arise due to the development of the new systems.
- It continuously reviews MIS development progress.

A. MIS ANALYSIS

The information system analysis addresses two main points: Conception and Preliminary Analysis. In the traditional system study, conception has been associated with the identification of a need which, supposedly, can be satisfied with the development of a system. This need can exist at any organizational level. Conception in the non-traditional study is explained in the MIS conceptual phase. The preliminary analysis is usually the first step taken toward the solution of the problem. An introductory study of the need,

and the practicality of using improved data processing techniques to satisfy it, are established during this phase. A discussion of these concepts in developing an MIS follows.

1. MIS Conception

The first important decision that the organization needs to make is whether the use of a computer will support and improve the decision-making process of the organization. Some of the factors which would justify a computer are:

- The need for handling large volumes of data more efficiently than with the system presently used.

- The need to perform complex computational problems, which cannot be handled efficiently by a manual.

- The need to perform repetitive tasks quickly, efficiently, and with accuracy.

- How the costs of manual systems, operating with large volumes of data and performing complex computations, compared with the costs of computerized systems.

- How the decision-making process can be improved through the use of computerized systems.

- How the strategic plans of the firm are affected in comparison with its computer-using competitors.

With this approach top management can determine whether to proceed with a detailed study. If a study is approved, the following steps will lead to the conception of an MIS.

- The creation of the steering committee and the establishment of the development team (see Appendix A).

- The steering committee will conduct a thorough study of the organization as it exists, and its history. This study will identify growth areas and pinpoint problem areas. Thus the need for organizational changes can be determined. A study of the goals of the organization will be conducted in conjunction with the functions it performs. The results to be obtained from this study can be summarized as follows: A clear definition of the objectives of the organization, a sharp definition of the functions performed, and the relationship between these functions. These functions (Finance, Personnel, Production, and the like) are the basis for the MIS.

All the necessary organizational changes are carried out during this phase. Placement of the Data Processing Service and the implementation of the Data Base Administrator function are also considered.

- The next step is to identify within each function those subsystems that can potentially be implemented on the computer. This is very rough approximation that will be improved later during the Preiminary Analysis Phase. Now the MIS has been conceived; a block diagram of its initial structure can be seen in Figure 3.1.

Figure 3.1 (a) shows the structure of a MIS according to the functions performed by the organization. Figure 3.1 (b) shows the subsystems division of each major system.

This approach of conceiving the MIS may have many detractors within the organization. Basically the arguments against this approach can be summarized as follows:

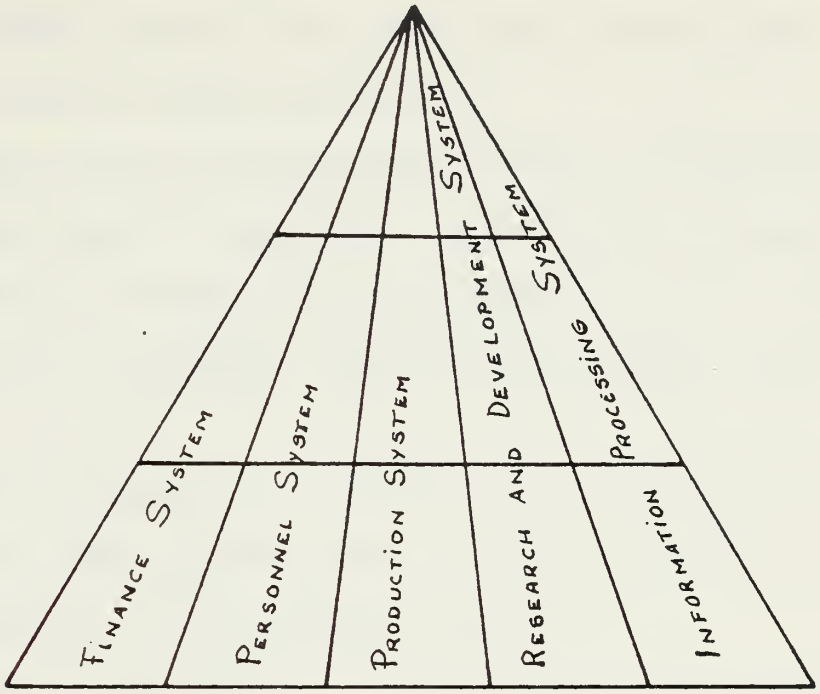


Fig. 3.1 (a) MIS Structure According to Functions Performed

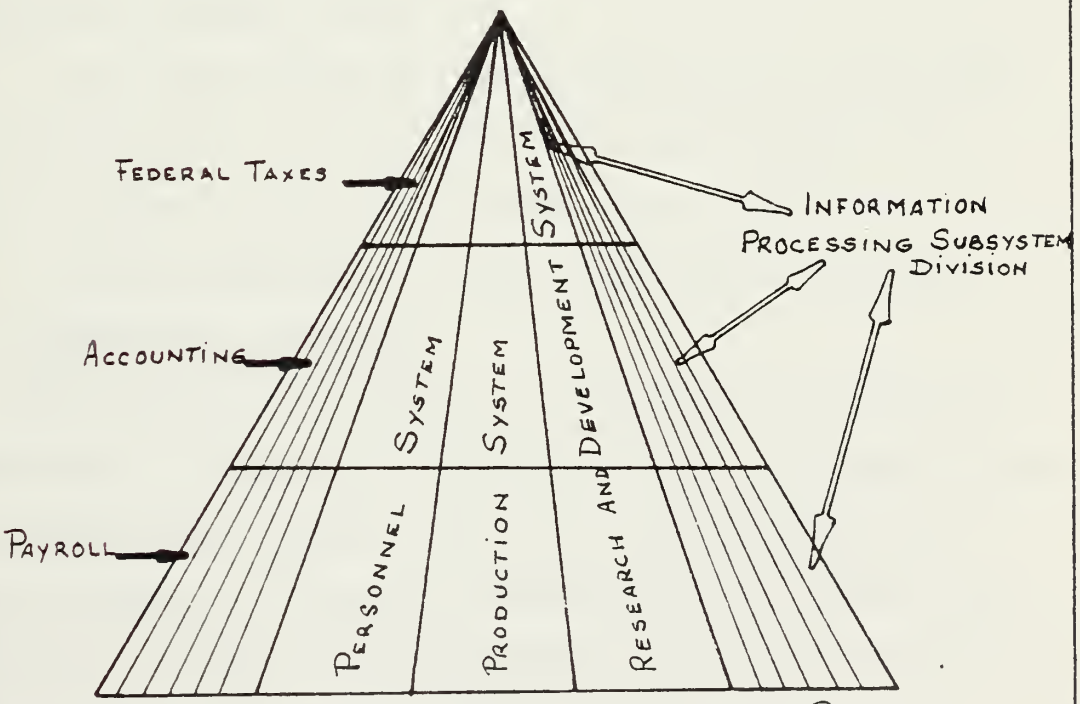


Fig. 3.1 (b) Subsystem Division of Each Major System

Source: References [5, 6, 11]

- It is a time consuming effort.

- It will divert, for some period of time, top management from its daily activities.

- The need for computerized systems is imposed on users who may question the need. As a result, a resistance to change may be encountered at different levels.

However, advantages can also be expected from this approach:

- Top management is a member of the steering committee; therefore their participation is assured at the earliest stages of the system development process.

- The participation of top management will encourage cooperation of their subordinates.

- This approach will provide a better planning for the system development effort.

- From the beginning, the potential use of the computer, at different decision levels, can be visualized.

- System integration is more easily obtained.

- System modularity is also more easily obtained.

2. Preliminary Analysis

The Preliminary Analysis is intended to accomplish the following: Prepare the Preliminary MIS development plan, identify user requirements, learn the present methods of handling information, identify problem areas, propose a set of solutions, conduct the feasibility study, prepare the implementation plan, and redefine MIS structure.

a. Establishment of Priorities

The steering committee will assign to each system that forms the MIS a priority for its study or analysis. These priorities should be given according to the importance of a system to the organization. Also, under the above concept, all the subsystems that are integrated into each system should be arranged by their priorities. The organization now has a first approach to the MIS development plan where systems and subsystems are organized in the way in which they must be analyzed, designed, and implemented.

b. Study of the Present Systems

The system development team will have the responsibility of conducting this study. Although there are some techniques that can be used for carrying out this analysis, the bottom-up approach (i.e., studying the actual systems starting at the operational level and going up to the strategic level) has been selected for the following reasons:

- Operational systems which are implemented can produce immediate benefits to the organization.

- Operational systems are easier to study than are tactical and strategic systems.

- The system development team will accumulate experience, improve its skill, and will get a better understanding of the system under study as it progresses up to the highest levels of the system. The system development

team will acquire a great deal of knowledge that will be useful in discussions with upper management.

- Most of the internal information that will flow through the MIS is generated at the operational level.

In order to make the study of the present systems easier the system is broken down into smaller structures called subsystems. Each one of these subsystems is studied by a system project team. This study covers the following points:

(1) Identify User Objectives. The user and system analyst project team will conduct a careful study of user objectives. The purpose of the system effort is to develop systems for the user; therefore, his objectives should be stated as clearly as possible. Some of the advantages of this approach follows [3]:

- Management will understand what is to be accomplished.

- The study team will have a well-defined set of goals.

- There will be a framework for evaluating the final outcome.

(2) Analysis of Actual Procedures. This study is concerned with the analysis of present procedures. Generally this analysis will accomplish the following:

- Identify the objectives of the existing procedures.

- Produce a diagram of the information flow indicating input sources, data concentration, processing, outputs produced, output distribution, organizations involved in the process, and the time associated with each activity.

- Highlight possible problem areas.

- Identify the costs and benefits of the procedures.

- Determine whether user objectives are satisfied.

(3) Redefinition of User Objectives. After studying actual procedures, the system project team can develop a better understanding of user requirements, and possibly, a redefinition of objectives will be necessary.

(4) Select Performance Criteria. The idea behind this activity is to establish the criteria by which the effectiveness of the subsystem is to be measured. This activity will continue throughout the development effort, becoming more detailed and refined with each stage. Some examples of criteria are cost and time savings.

(5) Identify Alternate Solutions. The system project team will establish a set of possible solutions for the subsystem under study, alternate solutions could be the following:

- Do nothing.
- Develop a manual system.
- Improve present system.
- Use service bureau.

- Use contractor.

- Develop an integrated manual and computer system (In-house).

- Develop a computer system (In-house).

(6) Identify Resources and Constraints. The system project team and the user will proceed to identify all the resources that are available for building the subsystem. Examples are manpower, budget, hardware, software, and other resources.

Constraints are all those limitations that will affect the subsystem development process. These limitations can be management directives, time required for implementation, budget, organization policy, and legal and regulatory documents [3].

Once the resources and limitations are established, a preliminary feasibility study can be conducted. This evaluation will consist of checking whether the proposed solutions can satisfy the user objectives and meet the performance criteria under the imposed limitations.

(7) Select Preferred Solution. The user and the system project team will study the alternate solutions and they will select the preferred one. The system project team will act as a council. Considerations about advantages and disadvantages of each alternative will be made clear to the user. Since systems are developed for the user, he will have the final decision in selecting the preferred solution.

(8) Prepare Proposed Subsystem. If other than the "do-nothing" alternative is chosen, the system project team will proceed to elaborate a draft of the proposed subsystem. The most important considerations for this study are the following:

- Make a complete and detailed statement of the user's requirements.

- Identify all subsystem outputs.

- Identify all subsystem inputs.

- Determine the necessary processes needed to produce the required information.

- After completion of previous steps, prepare a block diagram of the major functions to be performed by the subsystem and their interelationships to each other. See Figure 3.2 as an example.

- Identify all necessary files and their contents. This section of the study will be carried out by the Database Design Team. Both System Project Team and Database Design Team must have a close relationship throughout the System Development Process.

- Draw the information flow diagram for the proposed system.

- Draw the general flowchart for the subsystems.

- Identify resource requirements: Budget, Manpower, and Equipment.

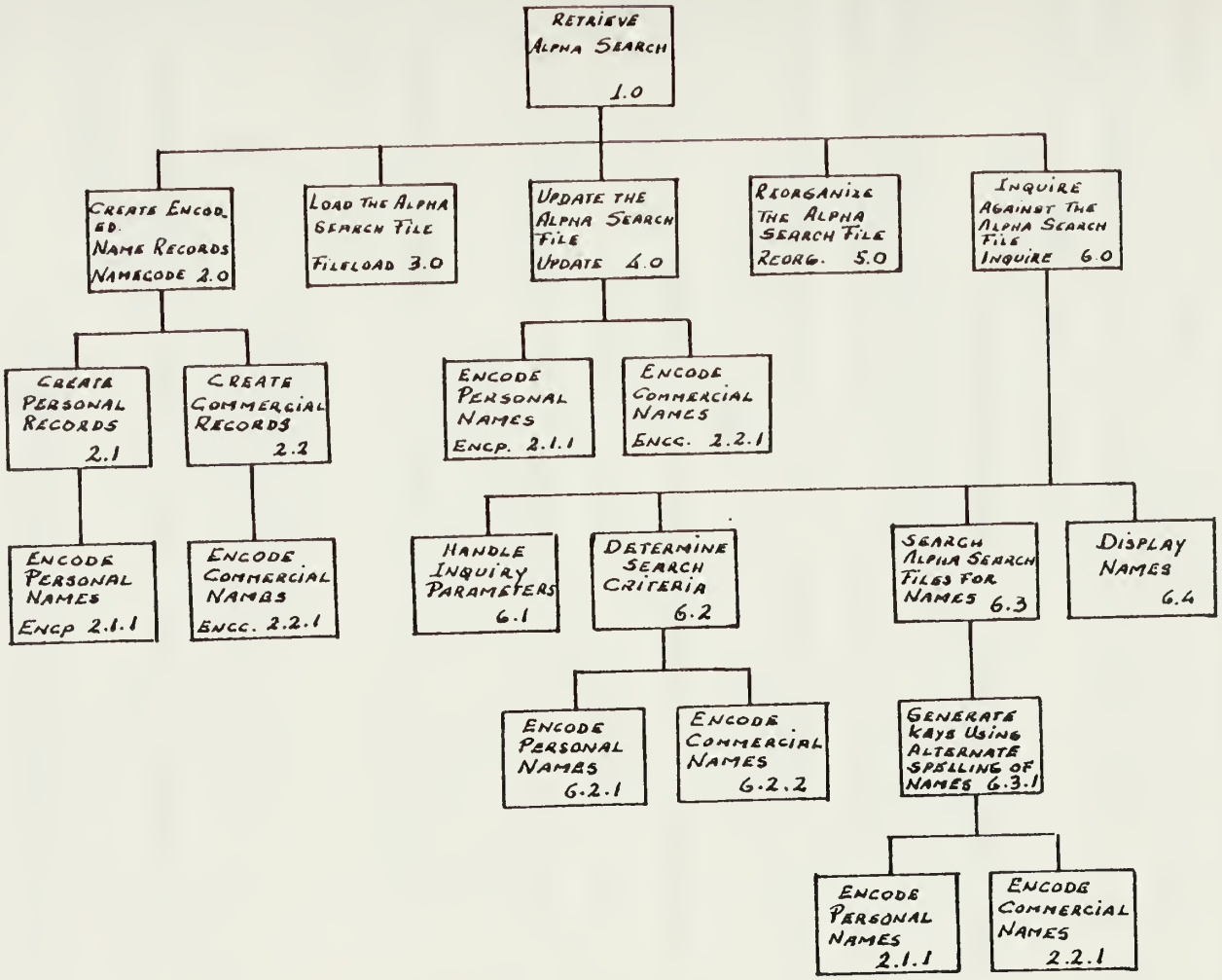


FIG. 3.2 EXAMPLE OF A BLOCK DIAGRAM OF FUNCTIONS PERFORMED BY A SUBSYSTEM

SOURCE: REFERENCE [36]

- Prepare an implementation schedule.

- Prepare a feasibility report.

(9) Write Functional Specifications. This is a description of the system, in terms of the functions it must perform. Input and output rates, response time, thruput, database (file) content, inquiries, and block diagrams are some examples of the specifications that need to be written during the preliminary analysis.

(10) The Relationship of Manager and System Analyst during the MIS Analysis. The responsibilities of Manager and System Analyst will now be discussed. The purpose will be to show what they should expect of each other during the different stages of the system development process. This guideline has been adopted from reference [17].

System Analysis is concerned with reviewing the existing system with emphasis on the constraints under which the present system operates. It is also concerned with defining the project and scope of the projects to be analyzed.

Manager's Duties

- His primary responsibility is to provide the analyst with a complete and accurate picture of the present system.

- Review present documentation with the analyst.

- Discuss information requirements.

System Analyst's Duties

- Make a thorough review of the existing system: how it works (documentation); how the users perceive it to work; and how it actually works.

- Be concerned with: input and output of present system; processing required to convert input to output; quality of present output; problem areas in existing system; policies under which existing system operates; interfacing of present system with other existing systems.

- Develop a complete list of documents and annotated: reports; forms; data source, volume, purpose, frequency, accuracy, and time lag of input to output.

- List of controls against errors.

- System flowchart (if necessary). Overall: an agreement between the system analyst and the manager regarding the operation of the existing system.

After examining the above list of duties it was concluded that the following items should be added:

Manager:

- Review and discuss with the analyst the selection of the performance criteria.

- Select adequate solution from the set of alternatives.

- Assist analyst as needed in preparing the proposed system.

Analyst:

- Prepare a rough draft of the proposed system.

- Prepare the feasibility report.

(11) Documents Generated by the Analysis. The feasibility report is the main document generated by the analysis. It contains summarized information of the previous activities. This document allows the steering committee to analyze and make its decision regarding future system developments.

The feasibility report is prepared according to the documentation standards of the particular organization.

The System Requirements Specifications is another report prepared by the system project team to specify requirements that a proposed subsystem must meet.

The Functional Specifications is a general document that broadly explains the components of the subsystem and what functions they perform.

In addition to the above reports all other pertinent information created to date are compiled and added to the portfolio of documentation of the analysis.

(12) MIS Refinement. The feasibility report is submitted to the steering committee for its review. From the set of reports that it has received to date, it extracts information about subsystems that were not considered in the initial conception, subsystems that will be cancelled, subsystems that will be necessary to combine, and the like. This information will keep the MIS updated with the information requirements of the organization. Also an arrangement of the priorities is considered.

B. MIS DESIGN

The overall idea in this phase is to provide a detailed design of the systems that will be implemented, establish software and hardware requirements, design the database, program and document the system.

1. Preliminary Design

The starting point of the preliminary design is the documentation generated during the preliminary analysis. Since the system development process is a matter of refinement, this looping-back to earlier stages of development is always present during the system life cycle. Evaluate Alternate Design Approaches, Select Preferred Design Approach and Write Preliminary Design Specifications, are the primary points treated in this stage [18].

a. Evaluate Alternate Design Approaches

Using as main input the system requirements and functional specification document, the system project team evaluates a set of alternate design approaches to carry out system requirements. Typical design approaches for a computer based information system will depend on the processing mode, the data organization, and the hierarchical approach. A general overview of these approaches follows:

By Processing Mode

- Batch.
- On-line (Also include real time and time sharing).

By Data Organization

- File Processing Appraoch.
- Data Base Approach.

By Hierarchical Approach

- Centralized Processing Systems.
- Decentralized Processing Systems.

b. Select Preferred Design Approach

After a careful evaluation of the alternatives is completed, the user and analyst proceeds to select the design approach that best fits requirements. Once again, the user has the final decision in this selection, but the system project team must have discussed with the user the advantages and disadvantages of the possible alternatives within the framework of the resources, constraints, and performance criteria. As an example, the design could be based on file processing approach, a decentralized computer system, and batch actualization.

c. Write Preliminary Design Specifications

The system project team, so far, has reviewed existing systems, proposed a new system, evaluated alternate solutions, evaluated different design alternatives, and selected an approach. The next step is to translate the design approach to a set of written specifications that describe the system. These are used to direct the efforts of the system project team and the users. The main activities that are undertaken in this phase are the following [19]:

- Identify master files, working files, data volumes, frequency of updating, retention time, data access methods and response time required from files (database).

- Determine the file (database) organization and layouts.

- Specify input rates, input layouts and the like.

- Define reporting requirements, volume, frequency and distribution.

- Determine controls and audit procedures.

- Identify computer programs and manual procedures required.

- Prepare program specifications.

- Identify hardware and software requirements.

- Revise estimate of operational costs of system.

2. Detailed Design

The main objective of this phase is to obtain a detailed system design for implementation. This means that the entire system has to be defined in terms of information flow database (files), inputs and outputs, procedures, program specifications, and test specifications. The main product of this phase is the deatiled system specifications. At this point computer programs can be written and the selection and evaluation of hardware and software can take place.

The following paragraphs indicate the main activities which are undertaken in this phase.

a. Identify Human Engineering Problem Areas

This activity involves identification of all areas of the system where human engineering is required to optimize personnel effectiveness, comfort, and safety. Work space layout, controls and displays, ambient conditions, training requirements, and task performance aids are examples of these areas.

The human characteristics identified here will subsequently be incorporated in procurement and design specifications of input/output equipment, communication networks, work facilities, job aids, training courses and other items related to the personnel subsystem. Some human factors considerations are: identification of critical human operations, specifications of aids to these operations, rules for man/machine interfaces, initial training guidelines and identification of organizational problem areas [3].

b. Develop Manual Forms

In this activity the design of all the manual forms that feed information to the subsystems are completed. In designing these forms care should be exercised to ensure that their design optimizes the function to be performed. The use of format design techniques and human engineering factors are considered. The key points to remember in the design of manual forms are: forms must be designed for each use, the form usage should be self-explanatory, all necessary information to fill out the forms must be supplied. This is normally written at the back of the forms.

c. Develop Reports

All the reports of the particular subsystem under study are developed at this stage. These reports are produced by the computer. The report design should be oriented toward use by a human being.

d. Designing the Database

The introduction of the database concept has changed the traditional system development process. At this stage, the project team designs the information files contents, organization, and access methods. However, in the database approach this design has taken effect at the beginning of the system life cycle. For this reason a more detailed study of the database design will be treated in Appendix B.

e. Define Systems and Audit Controls

As the project team has the design responsibility for those systems which it proposes, it also has responsibility for ensuring that the overall system will safeguard the organization by preventing the loss of operational capability due to internal system faults and hardware failure. The database control group is responsible for preventing deliberate destruction, accidental destruction, or corruption of data.

Some of the measures taken to protect against these failures are the following [19]:

(1) Physical Security. Physical security can be further subdivided in the following areas:

(a) Natural Hazard Protection. Covers fire, flood, storm and riot protection.

(b) Limitation of Personnel. Covers all the measures taken for preventing unauthorized personnel access to the computer facility. Strict controls are enforced when operating with sensitive data.

(c) Theft Protection. Measures taken for protecting data against burglary or lost. The use of TV cameras for monitoring the computer operations, the use of burglar alarms, and the like are typical examples of these measures.

(d) Hardware Protection. The parity check is an example of hardware protection. It can be said that they are check points built in the hardware itself.

(e) Computer Room Procedures. Deals with the procedures used for handling the operation of the computer room. Examples are the rules used for labeling tapes, and preventive maintenance of hardware.

(f) Library Control. Procedures used for controlling all program files, operating systems and utility software, as well as total system documentation. Only in this way can it be assumed that in a run the correct version of the program was used with the correct files.

(2) Systems Controls. This term is used to cover all those controls which are specific to a particular system.

Some of these controls will be included within the system itself, i.e. they can be done automatically by the computer, while others will be implemented manually by the user. The following list identifies some of the major system controls.

(a) Control Totals. A given data element is summarized as a separate clerical process and compared against the same summarized data element processed by the computer.

(b) Supervisory Checks. This concept deals with the monitoring of subordinates work by any level of management.

(c) Limit Checks. The basic principle is that the data is examined to see if the value is a reasonable value for that data item.

(d) Check Digits. The basic principle is that a reference number, like employee number, has a supplementary number added to it. This supplementary number has an arithmetic relationship to the rest of the number. This relationship can be checked by performing some arithmetic operations over the complete number.

(e) Verification. This concept deals with the verification of the input data before it is fed to the system. This is widely applied in installations that use keypunch machines or data entry systems.

(f) Sequence Checks. Certain data elements must occur in specific sequences and these sequences can be checked by program.

(g) Sample Checks. It is often useful, especially in the manual areas of a system, to institute a check on only a portion of the total data volume. It is similar to the quality control checks used by manufacturers in production lines.

(3) Audit Controls. These are a set of controls designed to meet legal and audit requirement. The basic requirement in this connection is the provision of a trail, called the Audit Trail, which enables the calculations performed on a certain item of data to be traced back through each processing step to the source.

f. Identify Computer Programs

Figure 3.2 shows the functions that the subsystems accomplishes. At this stage these functions are combined and/or subdivided into program descriptions.

Modular programming is used to segment the program into its logical parts or routines so that each routine may be programmed independently. Modularity allows independent testing of modules. There is a different set of factors that need to be considered depending upon whether programs are developed for batch or on-line processing. These factors as seen by Hice [3] follow:

The important factors to consider in defining batch programs are:

- Similar data base actions.
- Common logic requirements.
- Generic or specific similarities in outputs and inputs.

- Required software utility intervention.
- Specified program size.
- Intermediate output requirements.

The important factors to consider in defining real-time programs are:

- Input transactions.
- Processing sequence.
- Specific routine requirements.
- Output transaction requirements.
- Required software utility intervention.
- Logical processing pause or interrupt requirements.
- Specified program size.

Programs are first defined and then subdivided into a modular arrangement. This method produces program segmentation and facilitates load module development.

Also the selection of the computer language that is used for coding the programs takes place at this stage.

g. Develop Logic, Flowcharts and Tables

The broad information flow has been identified in the preliminary analysis and preliminary design phases. In the detail design phase this should be reviewed and broken down into a greater level of detail. A master flowchart of the subsystems is then produced, showing all inputs, files, processing stages, error recycling, and outputs of the subsystems. Care must be taken to identify the different time

factors. For example, frequency of reports, and length of retention of files and input data.

An important tool to use at this stage is decision tables. They are invaluable in defining the relationship between data, programs and users. Decision tables in conjunction with flowcharting are the major means of developing a system design. Advantages of decision tables are:

- They constitute a defined and orderly method of documenting analysis data.

- They are independent of the processing method, and may be used in any situation where decision must be made without regard to the availability of computers or other equipment.

- They lend themselves to automation because they use binary logic.

- The tabular approach to procedure definition aids the analyst in visualizing relationships and alternatives.

- Their rules of charting allow the detection of omissions and inconsistencies.

- Their simple format allows changes to be made easily and the effects of such changes to be easily recognized.

- Debugging is reduced because of the approximate error-free logic with which the final decision tables are constructed.

- There are rules to determine the internal consistency of decision tables. This is not true for flowcharts.

- Decision tables can be input to one of the decision table processors which has been developed, and it will produce coding in a particular language.

h. Write Program Specifications

The program specifications basically consist of the following documents [20, 21]:

- Diagrams of the preprinted input/output forms.

- Layouts of input and output records of the subsystem files with all data fields clearly identified.

- Master and detailed flowchart of the subsystem and its programs.

- Decision tables.

- Relationship between programs and subsystems.

- Information as to hardware, utility software, and programming language to be used.

- Processing characteristics.

- Test requirements.

The program specifications allow the project team to design, code, and test the subsystem programs. A set of specifications will be written for each program within the subsystem.

i. Develop Human Procedures

Human procedures are a series of statements that establish a course of action to be followed consistently under the stated conditions. The main idea behind this

concept is to increase work simplification and improve human effectiveness. All the procedures that will be carried out by the human being in connection with the subsystem will be analyzed and broken down into small pieces so that they can be minutely examined. This approach often causes an immediate visualization of areas that can be improved. Also associated with this concept are procedures for filling out input forms, distribution of reports, verification of data before automatic processing, and operations to be carried out in case of disaster or emergency. It is pointed out that the establishment of these procedures implies training for both the users and data processing personnel [3, 22].

j. Write the Subsystem Hardware and Software Requirements.

This step involves a complete analysis and revision of all the previous steps. The requirements for hardware and software for a particular subsystem will be summarized. These requirements are subsystem dependent, but basically the project team should establish requirements for [3, 23]:

- System processor(s).
- Amount of primary memory.
- Kinds and numbers of auxiliary storage devices such as printers, card readers, and card punches.
- Other equipment such as a system console, channels, hardware floating-point option, or storage protection for multiprogramming.

- Data communication facilities, such as terminals, multiplexors, concentrators, modems, front-end processors and switches.

- Operating systems.

- Languages compilers.

- Utilities.

- Data management systems.

- Software packages.

- Database systems.

This summary should be presented in a way similar to figures 3.3 and 3.4. They will be very useful for configuring the computer system for an MIS.

k. Develop Subsystem Test Plan

In this step the project team and the database control group write the test specifications. Basically the test specifications cover manual procedures testing, subsystem/system testing, acceptance testing, and hardware testing. Typical considerations at this stage are the following [3]:

- Identify the special software that will be used in support of the test which is not part of the system under test. An example is the use of test data generators.

- Give the location(s) where the test will take effect, the data(s), and participating organizations.

- Establish personnel requirements.

- Establish equipment requirements.

- Specify test objectives.

- Specify test extensions.

- Specify test constraints.

FEATURE	PAYROLL
FUNCTIONAL CHARACTERISTICS	
Processing Characteristics	Weekly, biweekly, semi-monthly, monthly, payrolls, hourly, salaried personnel, premium pay, overtime.
Source	Time card data, file maintenance data
Tax Considerations	Federal, state, city
Deduction Types	12 voluntary/employee; all have upper limits
Data Base Files	Payroll master file on tape; ancillary files on disc
Outputs	Payroll, paycheck, registers; time sheets; labor distribution, deduction reports; pay checks/statements
Other Functions	Deposits earnings in checking, savings accounts
Data Base Throughput	1,700 char/employee 1,200 checks/hr
OPERATING ENVIRONMENT	
Main Storage	32K bytes
Auxiliary Storage	2 disc (approx 15 mb), 3 tape (9-track, 800 bpi)
Input/Output	Card reader (1,400 cpm); line printer (1,100 lpm)
Source Language	COBOL ANS

Figure 3.3 Example of a Subsystem Summary
Source: Ref. [23]

TRANSACTION PROCESSING

FUNCTIONAL CHARACTERISTICS

Processing Characteristics	On-line production for the creation and update of files and directories for file/directory maintenance.
Input Sources	Manual keyboarding of full line entries or changes to line entries from 50 sources (terminals).
Frequency	Frequency of activity is constant from 30 terminals with remaining 20 terminals contributing at random intervals.
Data Base Files	Files and directories on disc.
Outputs	Call-ups of line items at remote terminal with edit responses to updates or changes.
Other Functions	Output of information into formalized reports and listings is performed in local batch mode.
Data Base	Approximately 180,000 accounts with an average of 1,500 char/account.
Response Time	Terminal response less than 10 sec.

Figure 3.4 Example of a Subsystem Summary for an On-Line Application
Source: Ref. [23]

TRANSACTION PROCESSING

OPERATING ENVIRONMENT	
Main Storage	Determined by number of simultaneously active users and operating system requirements.
Auxiliary Storage	300 million bytes disc storage for files and directories plus operating system requirements.
Input/Output	Remote teletype-like device or CRT.
Source Language	Conversational.
Communication Controller or Processor	Communication controller with ability to terminate 16 or more communication lines; performs line or page buffering, line multiplexing, and polling, as well as line control and interface functions.
Alternate	Depending on the other demands of the computer system, a small communication processor could be substituted in place of the controller, but for most intallations this would not be required.
Communication Line Adapters	Asynchronous line adapters for 1,200 bps polled (shared) private line communication circuits asynchronous line adapters for dial lines operating at 10 to 30 cps. All transmissions use the ASCII char set and teletype (model 33,35) protocol (line discipline).

Figure 3.4 (Continuation)

TRANSACTION PROCESSING

OPERATING ENVIRONMENT

Communication Lines	Six-private-line voice circuits hosting five terminals each; 10 in-dial standard voicegrade lines.
Remote Terminals	Teletype-like CRTs or keyboard printers with full ASCII char. set. Printers may be preferred over CRTs for off-line editing and documentation purposes.

Figure 3.4 (Continuation)

- Establish criteria for acceptance test.

This specification must be detailed for each subsystem and refined in the testing phase. The main products of this stage are the establishment of a subsystem test plan including test schedules and the creation of the appropriate test data.

User participation is crucial in establishing the criteria for acceptance of the test:

(1) Write Detailed System Specifications. System specifications are a set of documents that define the internal working parts of a new computer-based system. The major activities of this phase are as follows [24]:

- Work out a basic design for a programming system to support the user's operations.

- Flowchart the user's staff operations as they will be carried out under the new system.

- Develop final specifications for computer hardware and other equipment needed to develop and operate the new system.

- Finalize contract terms with vendors of software packages.

- Complete the design of programming subsystems, file and database structures, and program modules, including those needed for conversion of the user's current data files.

- Plan user documentation training.

- Ensure that internal auditors review the specifications, negotiate changes, and plan their own procedures for auditing the user's operations under the new system.

- Agree on criteria for accepting the system as operational, or terminating the development effort.

- Revise the cost benefit study.

- Review plans for the remainder of the project.

The final documents that are generated at this point depend on the standards established by the organization. However, the following list is a typical example:

- Detailed manual procedures.

- Flowcharts, tables and program specifications.

- Input and output specification.

- Final design of files (database), data parameters, and internal tables that will be shared by the programs within the subsystems.

- Final hardware and software requirements for a particular subsystem.

- User document specifications.

- Revised cost analysis document.

- Test specifications.

- Acceptance criteria.

- Subsystem implementation plan.

m. The Manager and the System Analyst during the Preliminary and Detailed Design

This guideline outlines the duties and responsibilities of Manager and System Analyst during preliminary and detailed design. It was adopted from reference [17].

Manager's Duties

- Develops an understanding of the systems analist's job, philosophy and attitudes.

- Should exert concurrent review over the output of the analyst.

- Primary duty is to review proposed system for adequacy.

Can be performed on a subsystem basis, but entire system should also be reviewed.

- This type of review should prevent major problems from occuring in the implementation stages of the system.

- Discuss proposed system with his staff with analyst present.

Systems Analyst's Duties

- Prepares the following documents:
  - System flowchart
  - Program flowchart
  - List of controls
  - Procedures manuals
  - Document design and layout
    - Input

- Output
- File design and layout
- Timing estimates
  - Conversion to new system
  - Run times
- Costs
  - Conversion to new system
  - Maintenance and up dating
- List of organizational changes
  - Job descriptions
  - Notify personnel department if union issues involved.

Overall: An agreement on what the proposed system should do, costs, involved, and its effect on personnel.

n. Documents Generated at the Preliminary and Detailed Design.

Although documentation will be treated later in this discussion as a step of the system development process, the reader has noticed that whenever applicable a reference to the basic content of a report, a manual, a procedure, or the like has been made. The idea emphasized here is that documentation is a continuous process that moves along with the system development process. Also, lack of documentation is one of the major problems in system development. The ideal situation is that the project team will not move ahead until all the documents of a step are completed.

The idea of presenting this discussion at the end of each main phase is to stress the importance of documentation and to identify the most important documents that should be generated at each stage. Preliminary and detailed design generate the following documents [25, 26]:

(1) Subsystem Requirements Manual. This is a description of what the subsystem is to be, what has to be done, and what is to be achieved.

(2) Present Subsystem Manual. This is an overall examination of the documentation generated to be sure that, in fact, agreement and full comprehension have been achieved as far as the existing subsystem is concerned.

(3) Proposed Subsystem Manual. Same as above but concerned with the proposed subsystem.

(4) Program Specifications. A detailed description of all programs. This document covers programs description, file or database description, processing logic and flowchart.

(5) Test Specifications. It is a description of the objectives of the test, expected results, and overall test plan.

(6) Detail Design Specifications. It specifies precisely how the subsystem will do what the functional specifications said it should do and how it will satisfy the requirements stated in the subsystem requirements specifications.

3. Configuring and Selecting the Computer Equipment

This section covers hardware/software configuration and selection process. Before going further in this study a background will be given in order to show what has been happening with the MIS development since the initiation of the study.

When the organization decided to move to computerized systems, a steering committee and a system development team were appointed.

The steering committee conducted a study of the organization and determined the major functions that were performed within the organization; such functions were the basis for identifying the potential systems that could be automated. Furthermore, each one of these systems was broken down in other small pieces which made up the potential subsystems within each system. The general structure of the MIS took form at that moment.

The next consideration of the steering committee was to establish a set of priorities for each system and subsystem.

The system development team came in and conducted a refined study of the potential systems and subsystems. The end product of this activity was the elaboration of the feasibility report and functional specifications. With the approval of the steering committee, the system development process continued and reached the point where the detailed design specifications were established. As part of this study, a document

called the Subsystem Hardware/Software Requirements was issued.

Although the term subsystem has been used throughout this discussion to mean a particular module of the entire MIS, it does not mean that the study of several other modules has not been undertaken in parallel. At some points the steering committee/system development team will consolidate all the information generated by the subsystems. This approach will be used several times throughout the System Development Process as a means of controlling system integration.

Selection and evaluation of the computer system is a point where the information generated by the subsystems must be consolidated.

The hardware/software evaluation team (see Appendix A), conducts this study. A thorough revision of the necessary documentation generated to date is carried out. Necessary changes are made as they are observed.

The evaluation team proceeds to summarize the hardware/software requirements of each subsystem. Possibly, the best way of doing this is by using matrices, where each row represents a particular subsystem and the columns represent specific requirements. At least two matrices are necessary, one for the hardware, and the other for the software requirements. Figures 3.5 (a) and 3.5 (b) show this idea. The figures were derived from reference [23].

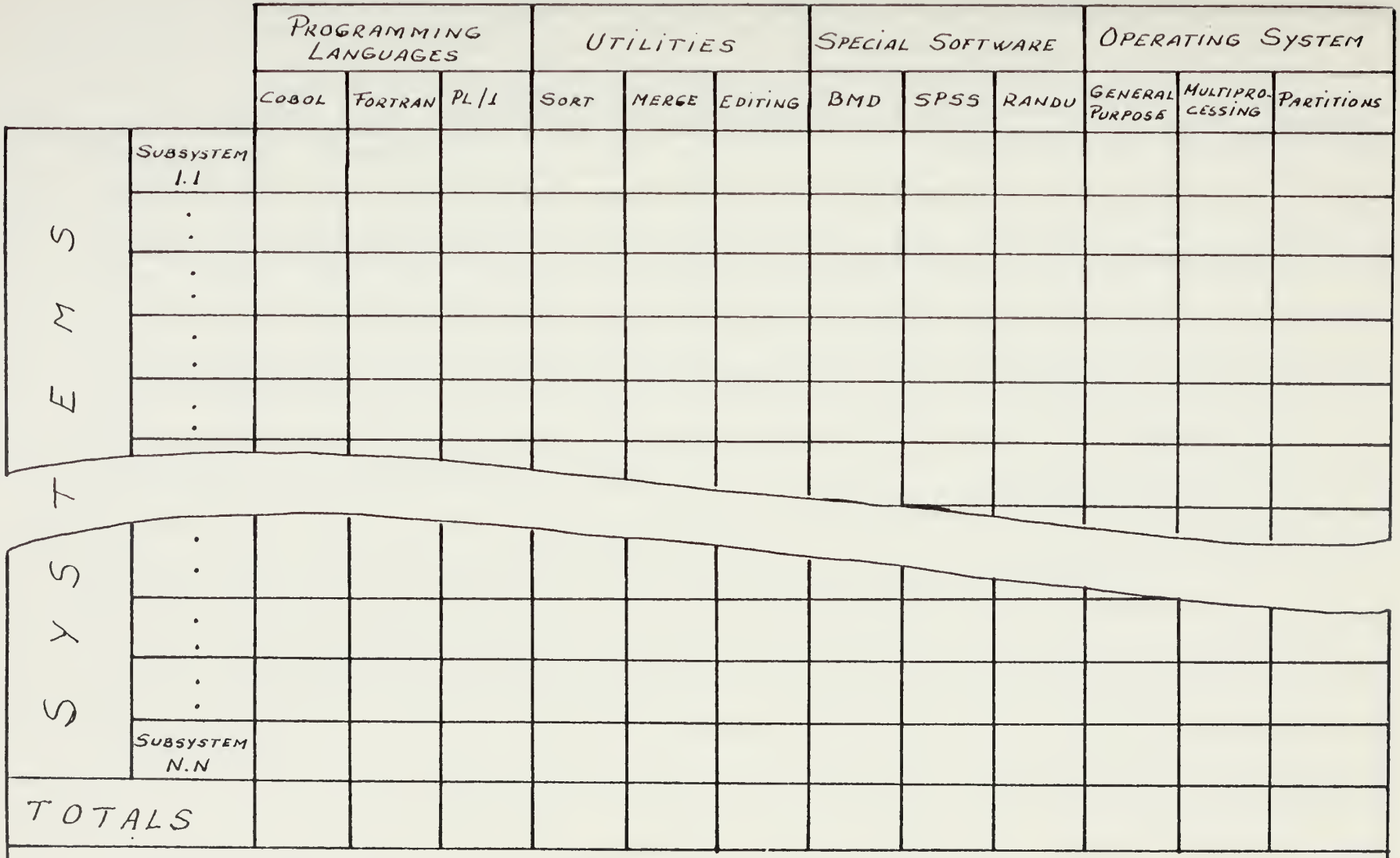


Fig. 3.5 (a) EXAMPLE OF AN MIS STRUCTURE REQUIREMENTS MATRIX

THE EXAMPLE DOES NOT SHOW ALL POSSIBLE SOFTWARE REQUIREMENTS

SOURCE: REFERENCE [23]

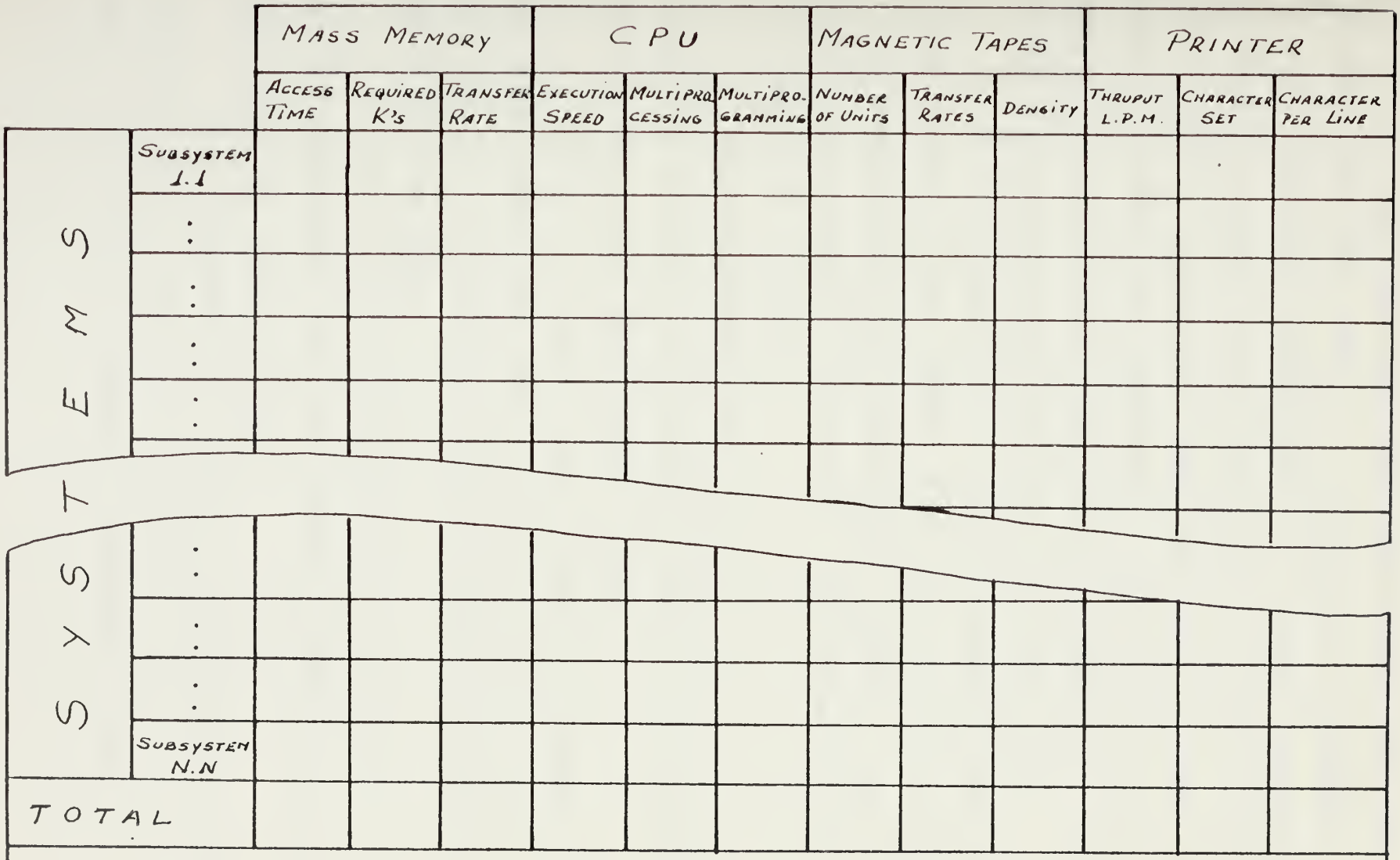


FIG. 3.5 (b) EXAMPLE OF AN MIS HARDWARE REQUIREMENTS MATRIX

THE EXAMPLE DOES NOT SHOW ALL POSSIBLE HARDWARE REQUIREMENTS

SOURCE: REFERENCE [23]

The next step is to study the matrices that have been created, delete the redundant requirements, and decide on the most reasonable alternatives. For example, several subsystems can specify the use of a printer for their outputs but it does not necessarily mean that several printers are needed to satisfy the requirements. One or two printers, perhaps, can satisfy the needs of the installation. Situations like this are analyzed by the evaluation team.

When the analysis is completed, hardware and software totals can be written. These totals are a good approximation to the hardware/software requirements. However, these requirements can be refined a little more. The following factors show why this is necessary:

- Not all design phase subsystems will finish simultaneously; therefore, many hardware/software requirements will not be ready at the consolidation stage.

- MIS is a dynamic system; therefore, hardware/software requirements will change through the life of the MIS, influenced by the growth of current applications and/or the addition of new ones.

- Establishment of requirements do not obey any mathematical formulation; they are a product of human estimation and, therefore, prone to errors.

The refinement will consist of extrapolating future hardware/software needs for a reasonable period of time. How much is reasonable is difficult to determine but the

organization cannot afford to have the system underpowered at the beginning of the production phase.

The evaluation team can proceed now to set up the computer system configuration that will support the MIS implementation.

Having established these requirements, the next step is to prepare the Request for Proposal (RFP).

The RFP contains the following sections [27]:

a. Management Information System Requirements

This section should contain the MIS requirements. This is a summarized information compiled from the subsystem requirements as defined during the Preliminary and Detailed Design phases.

b. Evaluation Criteria

In this section the criteria by which the proposals are to be evaluated should be explained to the vendors. This includes both the factors to be evaluated and their relative values. Generally speaking the factors that will be considered for evaluation may be grouped in Mandatory Requirements and Desired Features. Under Mandatory Requirements are listed all those items that are indispensable to meet the organization's objectives. Desired Features are helpful but not essential.

c. System Support

The kind and extent of vendor support necessary for attainment of all system objectives should be stated, and may extend beyond maintenance and training needs. Programming

assistance, special subroutines, documentation support, and other special requirements for which the user has a genuine need, may be herein defined.

d. Benchmark Data

The benchmark programs to be used should be supplied along with sample data.

e. Bidder's Conference Datas

The bidder's conference is a formal presentation to the vendor, by the user, of his system requirements. The date for the conference and a general explanation of its purpose should be included in the RFP.

f. Proposal Due Dates

The proposal due dates should be explained along with a clear description of what will happen to late proposals.

g. Vendor's Demonstrations and Presentations

Some time after the submission of the proposals, the vendors should be afforded the opportunity to explain their proposals. Having run benchmarks, the vendors should be required to provide demonstrations.

h. Contracting Conditions

The vendor should be informed that any promise he makes will be written into the contract and that the contract will have to be signed by a corporate official.

i. General Comments

This section should contain any instructions on the format of the proposal, such an arrangement of information

within the proposal, number of copies to be submitted, and the like.

The approach of this discussion is oriented toward the evaluation of a new computer system. However, other logical means could be used to satisfy the MIS requirements [28]:

- Use of a service bureau.
- Renting computer time.
- Purchasing a surplus computer.
- Use of time sharing services.
- Use of shared computer facilities.

A financial analysis of the above alternatives, considering their advantages and disadvantages, can be performed in parallel with the study for purchasing a new computer.

In parallel with the submission of the RFP to the vendors, the evaluation team must prepare the evaluation plan. This plan explains in detail, the steps and the considerations that are followed for evaluating and selecting the computer system. Basically, this plan covers the following aspects:

a. Proposal Evaluation

The proposals submitted by the vendors are examined thoroughly. The key points to look for in the proposal are completeness of the information requested, accuracy and validity of the information, ambiguities, ommissions or deviations from requirements, and proposals that do not meet the mandatory requirements. Vendors that do not satisfy

mandatory requirements are disqualified. Vendors which did not submit complete information are asked to supply it in a reasonable period of time. After that, the field of competing vendors should be narrowed to those who have been responsive to the requirements of the request for the proposal (RFP).

b. Hardware Specifications Evaluation

The evaluation team proceeds to study all the hardware specifications supplied by the manufacturers in the technical manuals. The purpose of this study is to determine if the offered systems meet the RFP specification. The following checkpoints may be considered when evaluating hardware:

- CPU (exeuction time, registers, addressing mode instruction set, memory capacity, multiprocessing, number of channels which can be attached, etc.)

- Channels (channels command, throughput, buffering, etc.)

- Mass Memory (capacity, buffering, access time, transfer rate, expansion modules, overlap, etc.)

- Magnetic Tapes (number of tracks, density, transfer rate, rewind speed, read backwards, etc.)

- Printers (character set, throughput, characters per line, etc.)

- Terminals (buffered/unbuffered, synchronous/asychronous, dumb/intelligent, programmable, human engineers features, transmission mode, special features, function keys, etc.)

- Communications Units (polling, transmission codes, transmission rate, control units, buffers, etc.)

- Simplicity of Operations.

- Expansion Capability.

- Reliability (can be given by the manufacturers, or can be obtained from system's users outside the organization).

- Back-up Equipments available in the localities close to the installation.

- Physical facilities required (space, air conditioning, electrical, floor ceiling, etc.)

- Delivery Schedules.

- Hardware Costs.

- Maintenance Costs.

- Technical Documentation Support.

e. Software Specifications Evaluation

This is a similar study as above but concerning the software. Basic checkpoints are the following:

- Operating Systems Features (schedule, memory management, I/O control, partitioning, multiprogramming, multiprocessing, interrupts, system generation, memory required, etc.).

- Languages Support (Cobol, Algol, Fortran, Assemblers, Report Generator, Simulation Languages, etc.).

- Utilities (sort, editing, debugging tools, errors diagnostics, file manipulation, etc.).

- Special software

- Satistical Packages (BMD, SPSS, etc.).

- Mathematical Packages.

- Database Management System (ADABAS, TOTAL, IMS, DMS, etc.).

- Compatibility (with other equipments and software).

- Telecommunications suitability (device handling, error processing features, re-entrant coding capability, bit manipulation, etc.) [20]

- Reliability.

- Software Documentation.

- Ease of Learning.

- Delivery Schedules.

d. Vendor Evaluation

A detailed evaluation of the manufacturers is conducted. Emphasis will be placed on the following points:

- Personnel Support (numbers and qualifications) [19].

- Vendor Support (maintenance, training, back-up facilities, technical assistance, etc.).

- Quality of Provided Documentation.

- Image.

- Vendor Demonstration.

e. Evaluation Methods

The technique(s) or method(s) that will be used for evaluating the computer system will be established in the evaluation plan. Some of these methods are listed below [18, 23].

- Simple and Sophisticated Mathematical Formulation.

- Hand Timing.

- Use of Benchmarks.

- Use of Simulation.

- Weights and Scoring.

- Minimum cost for fixed performance.

- Maximum performance for fixed cost.

- Cost Value.

- Present Value.

The first four techniques are used for validation of system timing. Through these techniques the evaluation team will ensure that the timing requirements of the MIS will be satisfy by the proposed configuration.

A brief explanation of some of these methods is presented.

(1) Use of Benchmarks. This technique involves the running of several applications which are representative of future computer workload on the vendor's equipment. The ideal situation would be to run actual applications but sometimes this is difficult to accomplish because they are being designed.

When actual applications are not available, the evaluation team uses models that simulate typical applications. Also, benchmarking programs can be found in the market, or can be adapted from other organizations.

The test consists of recording the computer run times of the application and uses the comparative times

of the application and uses the comparative times and resultant costs as a determinant in selecting a vendor.

It is not advisable to use only benchmarks for selecting a vendor because so many other factors have equal importance. This technique can be combined for example with the costvalue technique when making the final evaluation [28].

(2) Use of Simulation Methods. Through this technique the computer hardware, software, and applications programs are represented by a program model in order to develop performance and cost estimates for the computer workload on a variety of computer configuration [28].

This technique is costly and time consuming. Some simulators packages, available from manufacturers, are SCERT (Systems and Computer evaluation and review technique), GPSS (General Purpose System Simulator), CASE (Computer Aided System Evaluation), and CSS (Computer Systems Simulator).

(3) Weights and Scoring Method. In the weights and scores approach [28] the characteristics of a computer system are divided into major classes such as:

- Hardware
- Software
- Implementation
- Support
- Performance
- Expansion Potential
- Documentation, etc.

Each of these major clasifications are further subdivided until the degree of fineness desired is obtained. Now, the resulting set of characteristics describes the computer system. Each of these characteristics will be assigned a factor (weight) according to the importance that they represent to the information systems.

Also each major subdivision will have a weight factor assigned. For example hardware will have a weight assigned. It can be subdivided into CPU, Mass Memory, Printers, etc., each one of these subdivision will have their own weight factor. The CPU can be further divided into execution time, registers, number of channels allowed, etc., with a weight factor assigned to each one.

Each proposal is scored according to the degree in which they possess these characteristics. Now, the score is multiplied by the weight assigned to each characteristic.

The total of the scores of the characteristics of a subdivision are then multiplied by the weight assigned to that subdivision. The total of the scores of all subdivisions is then multiplied by weight for the major classification or division such as hardware. The resulting figure gives the total score for hardware. This procedure is repeated for all major classifications. The resulting score yields a measure of total performance score of the system.

Now, the cost of each system is considered, and it is divided into the total performance score. The

resulting figure is the Total Performance/Cost. This figure is calculated for each vendor and compared. The candidate with the highest figure is selected.

(4) Cost-Value Technique. In this technique, cost is the main criterion for determining the winner. The cost-value technique recognizes the necessity of evaluating the desirable features offered by the various computer systems proposed.

Simply stated, the cost-value technique amounts to taking the total cost of a system proposed and then deducting the cost values of all the desired extras included in that proposal. The difference represents the derived cost of satisfying the mandatory requirements stated in the specifications package. The system having the lowest derived cost for satisfying the mandatory requirements becomes the system selected [27].

f. Evaluation Schedule

This section of the evaluation plan will contemplate date, locations, and personnel that will participate in the evaluation process.

Whenever appropriate in this thesis some guidelines have been given showing the responsibilities of the users and the development team during the system development process. However, this time the presentation will be slightly different because several groups can be involved in this phase. The concepts were derived from reference [29].

(1) User Group Responsibilities. The user group is represented by the steering committee.

- Coordination of the activities that will be performed by the data processing, financial, and legal units.

- Negotiation with vendors.

- Development of feasibility study and cost/benefit analysis.

- Development of the acquisition study file.

- Selection of required configuration.

(2) Evaluation Team Responsibilities.

- Assist user group in negotiations with vendors regarding technical aspects.

- Evaluation and final approval of the hardware, software or service offered.

- Assist user group in preparing vendor's financial options to review by financial unit.

- Assist user group in planning the hardware/software implementation.

- Maintenance of a tickler file by date of expiration for lease and maintenance agreements.

- Prepare report of results of the evaluation and submit it to the steering committee.

(3) Finance Group Responsibilities.

- Analysis of the vendor's financial options.

- Arranging for and analysis of competitive bide for leasing funds, if leasing is the selected alternative.

- Determine tax benefits to be derived by the acquisition and which party can exercise these benefits.

(4) Legal Counsel Responsibilites.

- Review of all agreements before commitment by the steering committee.

- Recommendations of supplemental clauses to be amended to vendor's contract that will adequately protect the organization's interest.

- Participation in negotiations between vendors and steering committee on matters relating to contract law and special clauses to protect the organization's interest.

The final consideration in this stage is concerned with the documentation that is generated in order to support and document the evaluation process.

(5) Documents Generated During the Computer System Selection and Evaluation Study.

(a) MIS Hardware/Software Requirements. This document is prepared by the evaluation team and will consist basically of two two-entry matrices (one for hardware, the other for software) where rows indicate systems and subsystems that will run on the computer and columns indicate hardware or software requirements.

(b) Hardware Diagram. A schematic showing the different hardware components of the proposed configuration and their relationships.

(c) Software Diagram. As above but involving the software and the functions that will be performed.

(d) Evaluation Plan. Explained in the discussion.

(e) Request for Proposal (RFP). Explained in the discussion.

(f) Report of the Evaluation Result. This report will explain in detail the results obtained during the evaluation. Vendor proposal results will be presented ranked by some order, as for example descending order starting with the best result and ending with the worst one.

(g) Physical Requirements Document. Once the desired computer system has been selected the physical requirements for its installation will be established. It will cover space, floor, ceiling, air conditioning, power, costs, and other similar requirements.

4. System Programming

Programming is an activity that follows the detailed design phase. The input document to this activity is the program specifications developed in the detailed design phase.

The purpose of programming is to translate the program specifications into executable computer programs which may be tested.

In many system developmental efforts, the development of the manual procedures of the subsystem is overlooked and ignored in the shadow of the programming effort. This is a serious oversight because computer programs are only as effective as the manual processes which they support.

This stage covers the process of verifying the program specifications produced in the Detailed Design phase, producing detailed diagrams, coding the programs, documenting the programs, compiling and desk checking the programs, and finally debugging the programs to the point where they may be turned over to personnel responsible for system testing.

The system project team is responsible for this activity. Special care is called for in the case of multiprogramming and multiprocessing due to the increased complexity of the programming effort [3].

a. Problem Analysis

Problem analysis began in the Preliminary and Detailed Design phase. However, it has been said that the system development process is a matter of refinement; therefore, a careful review of the program specifications, before proceeding to the design phase, is a worthwhile effort.

What is expected at this time, is a good definition and understanding of the problem by the project team. This situation is covered by obtaining answers to the following questions [29].

- What must the program do?
- What are the inputs for which the programs must do its job?
- What outputs are required?
- What is the acceptable running time of the program?
- How much storage (in memory and mass devices) can be used?

- What other programs or operating systems must this program operate with?

- How does the program relate to the total environment?

- Which are the time and budget constraints?

- What standards or conventions are to be followed?

- Which hardware/software configuration will be used?

These and others similar pertinent questions cover the major decision points and allow the project team to start design. For each of these questions it is helpful to ask not only what is allowed but also what is not allowed.

Sometimes the user does not have a specific answer to a question, in this case the project team is free to design that element in any way, but further discussion, with the user, concerning the approach taken is necessary in order to make certain that it is acceptable.

All questionable areas found during the revision of program specifications are clarified with the user before proceeding with this phase.

The final product of this stage is the establishment of the program objectives.

b. Program Design

Once the problem is thoroughly defined, program design can begin. The project team reviews, one more time, the program specifications. Specifically, the program design approach which was previously established in this document. The steps to be reviewed are the following:

- Identify major functions that the subsystem must accomplish.

- Identify subfunctions that will be performed for each major function. Divide subfunctions in small modules as necessary, according to the degree of fineness required.

- Identify function, subfunction, and modules relationship.

- Review the block diagram that indicates functions, subfunctions, and modules relationships. Generally speaking, top-down design and modularity have been used when performing previous steps.

- Review flowcharts.

- Review tables and decision tables.

- Review completeness of the documentation of previous steps.

- Review of the final product - the program specifications.

All the necessary corrections are performed before continuing to the next steps: coding, debugging, and documentation.

c. Program Coding

Program coding should be structured and performed according to pre-established standards. Readable and efficient coding is learned only through continuous practice writing programs in a disciplined way. For this reason guidelines that may help in writing programs will be offered in this section. These guidelines are the following:

- The design must be completed before any code is developed.

- Project team (or individual programmer) should read carefully the program specifications, become familiar with them and discuss extensively and in detail any point that has not been made clear.

- Project team (or individual programmer) should be familiar with the chosen programming language, operating system, utilities, and hardware configuration where the program will run. This point is difficult to accomplish when the organization is planning for its first computer.

- Project team (or individual programmer) should choose a coding technique prescribed by the installation. Top-down coding and bottom-up coding are the most used techniques.

In top-down, coding is generated starting with the main program and then going down step-by-step to the lower levels of the program. These lower levels may be subroutines, nested subroutines, functions, and priruitives.

In bottom-up, coding is generated starting at the lower levels, progressing up to the higher levels, and finally to the main program. Top-down coding is the preferred method.

- Programs should be written with simplicity in mind. Complex coding techniques should be avoided whenever possible.

- Machine independent programs should be written whenever possible in order to assure portability; however, efficiency may be reduced.

- Weinberg [30] suggests that it is a good practice to encourage each programmer to present his program design and coding, to all members in the same project team, for their review and criticism.

- Programs should not be data dependent. Generality allows easy modifications and leads to fewer errors.

- Programs should be fully documented, internally as well as externally. See the section on Program Documentation for more details.

- The use of structured programming and indentation makes program listings more readable. In a time-sharing environment indentation techniques can be automated.

- Programs should be desk-checked before submission to the computer.

- The inclusion of controls at strategic places in a program should be the practice. Control totals, check-digits, data boundaries control, detection of invalid data, such as using alphabetical where numerical should be used, and the like are examples of controls that may be included in the coding.

This practice is a useful tool in debuggin and testing.

d. Program Debugging

"Debugging is an art: the art of finding the nature and location of an error once its existence has been established." [31]

Debugging is sometimes confused with testing whose objective is to show the presence of errors in a computer program.

Like coding, debugging is also a matter of experience and continuous practice. Experienced programmers can fix "bugs" in a program more easily than novice programmers. The term bugs, just introduced, is a synonym for errors.

The purpose of this section is to present some of the methods and philosophies used in debugging programs. Sources of error, debugging tools, and planning of debugging runs are some of the most important topics covered below.

(1) Sources of Error. A large percent of errors that appear in a program are introduced by the person responsible for its coding. All programmers, ranging from experienced to the novice, can introduce errors in a program.

It can be said, that bugs have been passing from generation to generation of programmers. Errors committed by programmers of the 1950's are still committed by programmers of the 1970's. It is a good practice in organizations to keep a record with historical information about this matter. Identification of the errors, what caused them, and how they were fixed, is an invaluable reference in a data processing organization.

The following is a list of come of the common sources of errors that appear in computer programs [31, 32].

(a) Keypunching and Clerical Errors. In this category fall all those errors originated during the

transcription of source documents (as coding sheets) into cards, tapes, or other recording media. Confusion between letter Z and number 2 is an example of this category.

(b) Uninitialized Variables. Variables that are used before being initialized. Some compilers have features that set up all the uninitiliazed variables to a predetermined value at compilation time. Normally, they are initialized to zero.

(c) Sharing Variables Among Many Modules. Two or more modules trying to use the same variable or memory location for their temporary storage.

(d) Control and Logic Errors. These errors are difficult to find and also difficult to explain. Branching to the wrong labels and calling the wrong modules are examples of this category. Generally speaking, in this category fall all those design failures which the programmers, consciously, wish not to happen.

(e) Array Subscripting Out of Bounds. It happens when during computations the program begins to address before the start or past the end of the assigned array memory area.

(f) Improper Arithmetic Operations. During computation division by zero or negative squared root are not allowed. These are eamples of improper arithmetic operations.

(g) Syntax Errors. This error happens when the program statements do not conform the syntax rules established for the programming language.

(h) Data Errors. Normally, these errors happen when provisions are not taken to anticipate the ranges of data. A mathematical operation performed on an alphabetic piece of data is a typical example.

(i) Missing Program Cards, Out of Sequence Program Cards, Additional Program Cards. Errors due to these causes can happen when careful handling of card decks are not observed.

(j) Using the Wrong Versions of the Program. The use of out-of-date versions of a program may be a source of errors.

(2) Debugging Aids.

"Debugging aids are stethoscopes necessary to isolate the cause and location of an error." [32]

Debugging tools, debugging aids, and debugging techniques are synonums. They can be described as a set of tools used by the programmer in order to facilitate the debugging process, i.e., the process of finding and locating errors. The most typical debugging aids are: dumps, traces and dynamic debuggin techniques (DDT). An explanation of them follows [31, 32]:

(a) Memory Dumps. In this technique the contents of core memory can be printed at any time during the execution. Normally, the output is given in a hard-copy device like a printer, but outputs to tapes, or disk are also possible and printed later.

The information obtained concerns the status of a program at a given time. This information is printed in machine code and requires the programmer to understand machine language and to be able to relate the machine language to his high-level language program.

(b) Traces. This is another technique very useful in debugging. Traces can be implemented by the programmer in the structure of his program or can be a part of the software package supplied to the installation. This technique causes some portions of the memory to be printed at times, or under conditions specified by the programmer. A trace can be used to see if the program is executing in the sequence in which it was written, and to see if the variables have the desired values stored in them.

(c) Dynamic Debugging Tools (DDT). This is a more powerful tool than memory dumps and traces. There are different kinds of DDT packages that can be implemented. They are the following [31]:

- Stand-alone DDT is one of the simplest and is often used to debug programs on small machines.

The main characteristic of this package is that it handles its own terminal communication with the programmer. The I/O communication is handled without the intervention of the operating system.

Stand-alone DDT does not allow the execution of other processes while it is in use. Also, it does

not allow the programmer to examine or modify a program while it is running.

- Time-sharing DDT differs from stand-alone in that the terminal communication is handled by the operating system. This means that the operating system can perform other functions while it is waiting for the programmer to provide terminal input to the time-sharing DDT. However, the program that has been debugged cannot run concurrently with the DDT.

- Real-time DDT is the most complex form. It allows other programs to execute while it is waiting for input from the programmer's terminal. The program being debugged and DDT can run concurrently; therefore, modification or examination of the program is possible while it is executing.

(3) Planning a Debugging Run. The idea of planning a debugging run is concerned with finding and fixing the bugs with a minimum expenditure of programmer and machine time.

A typical debugging plan may cover the following steps:

(a) Desk Checking. Desk checking begins with desk (card deck) checking. A careful examination of the card deck will minimize the possibility of missing, out of sequence, or extra cards.

Programmers who prefer to punch their programs should have them verified by a keypunch verifier.

The next step is to produce a listing of the source deck. The listing is compared against the coding sheets and to the flowcharts instruction by instruction. At this step the programmer will try to find differences between the logic of the code and the flowcharts. If modular and/or structured programming were used their value would be apparent at this stage.

Whenever possible, the programmer should submit his code and flowcharts to somebody else on the project team for review and criticism.

(b) Test Data Cases. The next step is to prepare a set of data that can flow through every path of the program at least one time. Since a program may be sensitive to certain data values as well as data ranges test cases should be established for a variety of values [29].

The test data cases are prepared considering valid input data values to the program and also data outside the range of permissible values to test the error detection capability of the program.

Test data cases can be obtained from the users, asking other members of the project team to prepare them, or through the use of test data generators.

(c) Controlling the Debugging. The debugging process should be conducted in a controlled fashion, rather than in an indiscriminate manner. The following steps should be followed for successful debugging:

- Programs should be run for the first time with a minimum amount of data. The purpose of this procedure is to clean the programs of syntax errors, keypunch errors, and the like.

- Once a program has been cleaned, it can be executed with the test data cases to see how the program behaves. Special attention is placed on verifying the relationship between a set of inputs and their respective outputs.

- All the program listings produced during debugging are carefully marked and filed for reference. The idea is to have historical information of all the steps taken to correct the bugs.

- The last consideration is that the programmer should be familiar with the computer system and software packages, specially those parts of the software useful in debugging, and with computer room procedures.

e. Program Documentation

The process of documenting the program starts during the preliminary and detailed design. Some of these documents have been explained throughout this discussion. These documents cover functional block diagrams, flowcharts, tables, decision tables, files and database organization, input/output forms, manual and computer procedures, and all other aspects of the program.

The documents referenced above can be considered as the external documentation of the program. Programs can

be also documented internally. This technique is called intraprogram documentation.

All the programming languages, including assemblers, allow the insertion of comments within the program. Intraprogram documentation, basically, should cover the following:

- Program identification, either by name or encoded form.
- Version of the program.
- Creation date.
- Originator's name.
- Brief description of the purpose of the program.
- For each subprogram, subroutine, or function, a brief description of its purpose.
- A listing of all the mnemonics that identify variables with their full names and a description of their use.
- Two or three statements describing important processes or calculations.

The insertion of comments in a program do not reduce its efficiency because they are not considered at execution time.

The use of indentations when writing programs produces neat and readable documents.

There are other documents derived from a program, user's manual and operator's manual, but they are discussed in the documentation phase of the system development process.

5. Documentation

> "Documentation management is a demanding task. To make use of a well known analogy, final system documentation is much like the weather: Everyone talks about it but no one does anything about it." [33]

By this discussion, a section dealing with the documents that must be generated at each step has been inserted. If the documentation is generated at each stage during the development of the system, there is no need for a final system documentation phase.

The major portion of a MIS development is the publication of documents. From the beginning the project team issues a stream of forms, reports, memos, flowcharts, and diagrams. Each of these must be prepared, reviewed, revised, circulated, revised again, and maintained. Later on computer programs will appear in the project and they create a need for still more documents.

All of these plans, specifications, memos, and file descriptions document a project. They establish the reasons for starting the project, record decision points made during development, reflect changes in plans, and after months, and often years of work, only a few describe the complete computer system [34].

Documentation is the only mean through which the communication between a computer based-system and personnel who operate, maintain, manage, or audit it, can be established. The preparation of good documentation unambiguously defines the following:

- Purpose of the system or program.

- Procedure for running it.

- Format and flow of data through the computer system.

- Controls included to ensure the validity and integrity of the data being processed as well as the results.

- Logic of the operations that the computer and humans perform.

- Inherent limitations and assumptions.

a. Problems Resulting from Inadequate Documentation

Some of the specific problems caused by inadequate documentation are summarized below [35].

(1) Programs Must be Rewritten. When it is necessary to modify or to change a program, the absence of documentation will make going through the listing program, understanding it, and figuring out the possible changes, almost impossible. Even for the original programmer it is hard to understand his program after a few months. The result is that modifications that could be easy to implement will cause a program to be rewritten for lack of documentation.

(2) Systems Must Be Redesigned. The consideration is the same as above, but now several programs may need modification. If documentation does not exist the whole system must be redesigned.

(3) Excessive Time to Make Modifications. The amount of time required to go through the program listings can be long or even unsuccessful, after wasting considerable amounts of time due to the lack of documentation.

(4) Difficult Auditing Process. Lack of documentation makes it difficult for auditors to identify internal controls provided in a system; therefore, system audits may not be accomplished.

b. Causes of Inadequate Documentation

In the system development process there are many factors that may cause inadequate documentation. When adequate controls are not exercised, designers and programmers will devote their efforts to have a working system as soon as possible. After the working system is obtained they start worrying about documenting the project; but the "paperwork" involved in this activity is so overwhelming that they will never finish the documentation of the system. If documentation is completed the system will probably be poorly documented because the designer or programmer lacked time and must recall many things from memory.

The other reason for inadequate documentation concerns enforcement. If standards are established, management must establish some control points during the development activity in order to ensure that the necessary documents are generated in the appropriate stages. Management must not allow the project to proceed to other phases of development if the appropriate documents were not completed in the preceeding phase [35].

Some approaches to the solution of the documentation problem are the following:

- Establishment of documentation standards.

- Management review and enforcement.

- Establishment of technical writing staff who will produce well-written documentation from the manuscripts of programmers and system designers.

- The use of design and documentation techniques like HIPO (Hierarchy plus Input-Process-Ouput), developed by IBM. A detailed description of this technique can be found in reference [36].

c. Documents Required in the System Development Process

This section will present a summary of the most important documents that will integrate the documentation portfolio of a MIS; each one of the referenced documents is necessary for each subsystem that comprises the MIS; these documents as quoted in reference [25] are the following:

(1) Analysis and Feasibility Study Report. This is a proposal to management requesting approval of a new system and the budget for it.

(2) System Requirements Specification. This specification is prepared by the system designer to specify business requirements that a proposed system must meet.

(3) Functional Specifications. This is a general interim document that broadly describes what the system will consist of and what functions it will perform.

(4) Detail Design Specifications. This is developed from the functional specifications. It specifies

precisely how the system will do what the functional specification said it would do and how it will satisfy the requirements stated in the system requirements specification.

(5) Program Specifications. These are prepared for each program or processing entity that is a component of an application system.

(6) File or Database Specifications. These are prepared for each file (or database) that is a component of an application system. They specify the fields (or data elements) and records that make up the files.

(7) Test Specifications. These specifications document the requirements and results of program, subsystem, system, acceptance and parallel testing.

(8) System Maintenance Manual. This manual provides a master reference and cross-reference to all system components.

(9) Computer Processing Specifications. This specifies all the job processing information required by computer operations for the routine processing of every system job.

(10) General Information Manual. This provides a system summary for executives and top system-user management.

(11) System User's Manual. This manual is used to train user personnel and as a reference source for solving simple operational problems. It is primarily intended for supervisory, administrative and clerical personnel in user departments.

(12) System Procedures Manual. This manual provides step-by-step instructions for all activities, jobs, and tasks associated with the system.

(13) Terminal Operators Manual. This provides instruction for the support of dedicated use of on-line terminals.

If all the documents reference above, and other miscellaneous documents, are generated in the appropriate phase of the system development process according to the standards of content and quality established by the organization, the final product will be a well-documented system.

C. MIS IMPLEMENTATION

In this stage all the necessary activities for making the developed system operational and shifting control to the user are carried out.

Testing, implementation, operation, maintenance and follow-up and system cessation are the main topics discussed in this stage.

1. Testing

Testing of computer programs is a frequently ignored area. There is little appreciation for the scope of the problem, to the point that not even among the data processing people is the concept of "testing" universal [31].

The objective of this discussion is to present an overview of the current techniques and strategies that can be applied to testing.

The key question in testing is the following: Given a computer program, how can the programmer determine whether or not it will do exactly what it is supposed to do?

This section attempts to answer this question. Basic definitions, scope of the testing problem and testing methods are the main points treated in this discussion.

a. Definition and Concepts

Going back to the preliminary steps of the system development process, the reader will recall how the MIS was structured. Figure 3.6, shows a block diagram of the structure of the MIS parts by levels. Each one of these levels indicates where testing needs to be performed. Level 0 (zero), at the bottom of the figure, shows the lowest and simplest unit of testing in an MIS, the module. As the testing team progresses up in that diagram the units of testing are relatively more complex, and therefore, more complex to test. The following are a set of basic definitions and concepts that form part of the testing terminology used throughout this discussion. These concepts were quoted from references [31, 32, 37].

(1) Module. A module is a collection of instructions sufficient to accomplish some logical function. It is sometimes defined in terms of physical constraints, such as the idea that a module should not consist of more than fifty statements. There is also an almost general assumption that a module is not normally useful unless it is combined with other modules to form a larger self-sustaining entity (i.e., a program).

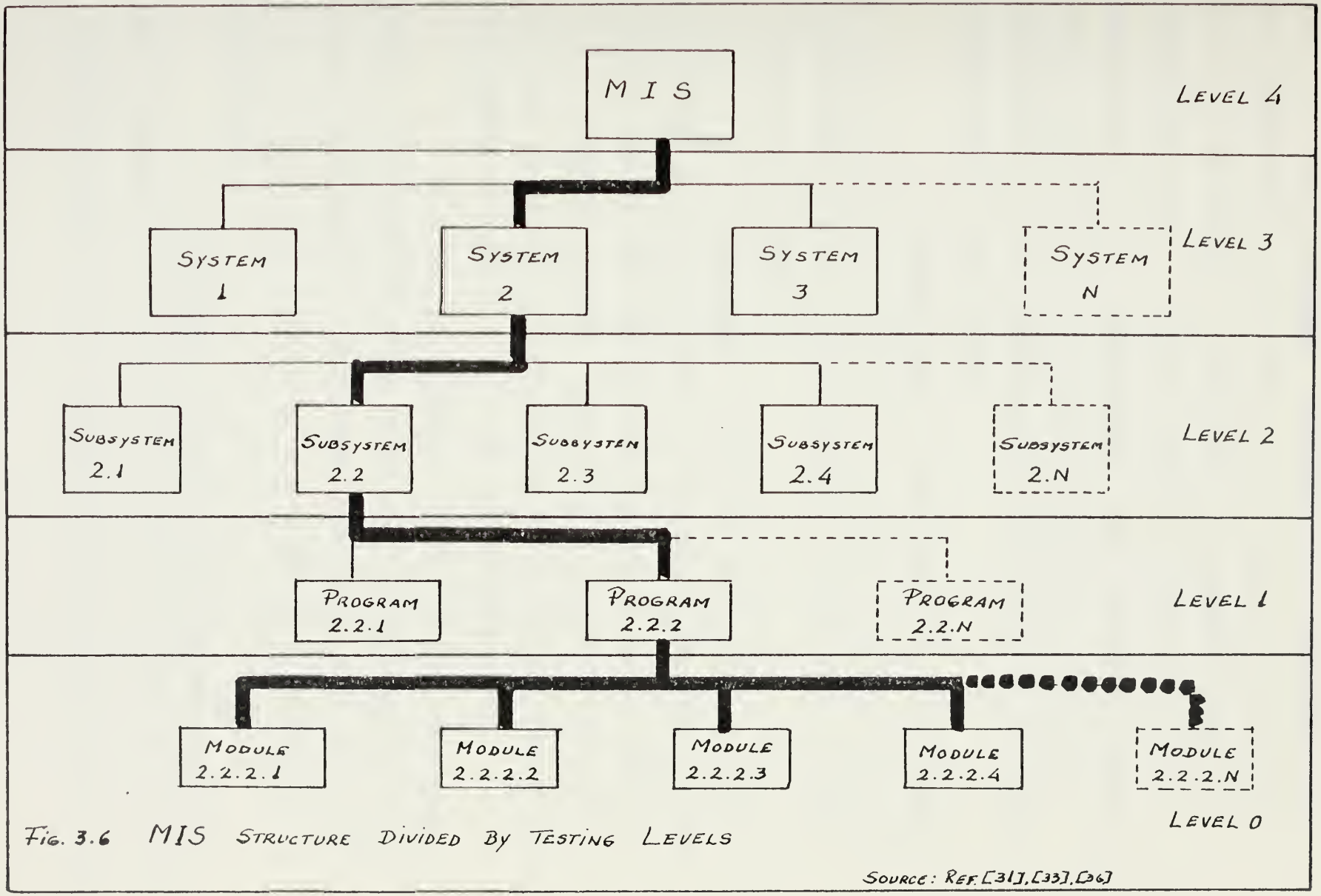


Fig. 3.6 MIS Structure Divided By Testing Levels

Source: Ref. [31], [33], [36]

(2) Program. A program is a collection of instructions which, when combined with appropriate data and control information is sufficient to accomplish some well defined function. A current tendency exists in the data processing field to form programs from a different number of modules that when put together can perform a meaningful function that is the combination of the isolated functions accomplished by each module.

(3) Subsystem. A subsystem is a collection of programs organized in order to accomplish a larger and more complex function that would normally be done with a simple program (i.e., payroll).

(4) System. A system is considered to be a collection of programs and/or subsystems sufficient to accomplish a significant coherent application or major function (i.e., personnel system). This word is often used to describe any collection of programs provided by the vendor with the computer hardware (i.e., the operating system).

(5) Bug (or Error). A bug, in the programmer's jargon, is a software error which can be repeatable and consistent or transient. This term is normally used when referring to logic errors, instead of syntactically incorrect statements.

(6) Testing. Testing is the process of executing a program with the intention of demonstrating conformance to specifications. It is pointed out that during testing the

main idea is to stress the program so that if errors are present, it will fail.

(7) Proof. A proof is an attempt to find errors in a program without regard to the program's environment. Assertions are established about the program's behavior and then mathematical theorems about program correctness are derived and proved. A proof does not involve program execution in the computer.

(8) Verification. A verification is an attempt to find errors by executing a program in a test or simulated environment.

(9) Validation. A validation is an attempt to find errors by executing a program in a given real environment. Verification and validation are often used interchangeably, but they are not the same.

(10) Module Testing. Module testing involves testing an isolated module in order to establish its correctness.

(11) Program Testing. It is the process of trying to discover errors that appear when all the component modules of a program are put together and executed. The great majority of errors discovered in this phase are caused by poorly defined interfaces between modules.

Depending upon whether the testing is carried out in a simulated or a real environment, it will be a verification or a validation, respectively.

(12) Subsystem Testing. Testing of several programs which constitute a subsystem.

(13) System Testing. Testing of several subsystems.

(14) Integration Testing. It is the verification of the interfaces among system parts (modules, programs, and subsystems). It can also be referred to as the process of verifying the interfaces between systems that form an MIS.

(15) Retesting. It is referred to as redundancy testing and regression testing. After an error is discovered and corrected, in any phase described, the retesting is conducted again in order to ensure that the errors were removed and no other errors appear because of the correction just made. Note that this process can be applied as many times as necessary.

(16) Acceptance Testing. It is the testing of the system or program to user requirements. Normally, the user establishes the criteria for accepting the product.

b. The Scope of the Testing Problem

The introduction stated that testing is one of the less developed fields in data processing and that people in the computer field do not have a universally accepted definition of testing. Most programmers, many programming managers and almost all non-technical managers drastically underestimate the amount of time, energy, planning and resources (money) that should be devoted to testing. This also holds for debugging and maintenance. To illustrate the

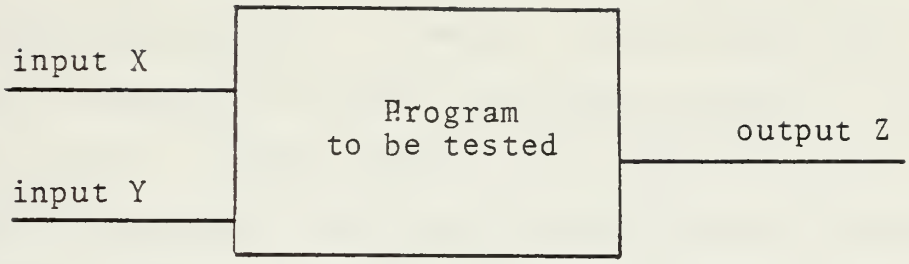
complexity of the problem, some of the statistics that were found during this research are presented [31]:

- The amount of time required for adequate testing of programs varies from 30 percent of the total project to 50 percent or even higher.

- Eighty percent of the total money expended on the NASA Apollo project was devoted to various forms of testing. However, there are some schedules that call for two years of design and implementation and only one month of system testing.

- The number of bugs remaining in large problems, after they have supposedly been thoroughly tested, is rather large. It has been estimated that each new release of OS/360 contains over one thousand (1,000) errors. Besides, one EDP consultant is Oslo, Norway, claims to have counted over eleven thousand (11,000) bugs in a recent release of O.S.

Other consideration that need to be studied include the amount of testing necessary to carry out prior to the acceptance of a system. Logically, the testing team should try, during a test, to exercise all the possible paths during the execution of a program and to have all the possible input combinations represented as test cases. As an example, analyze the following diagram: Suppose the program to be tested has two input variables and one output variable, as depicted below [38]:



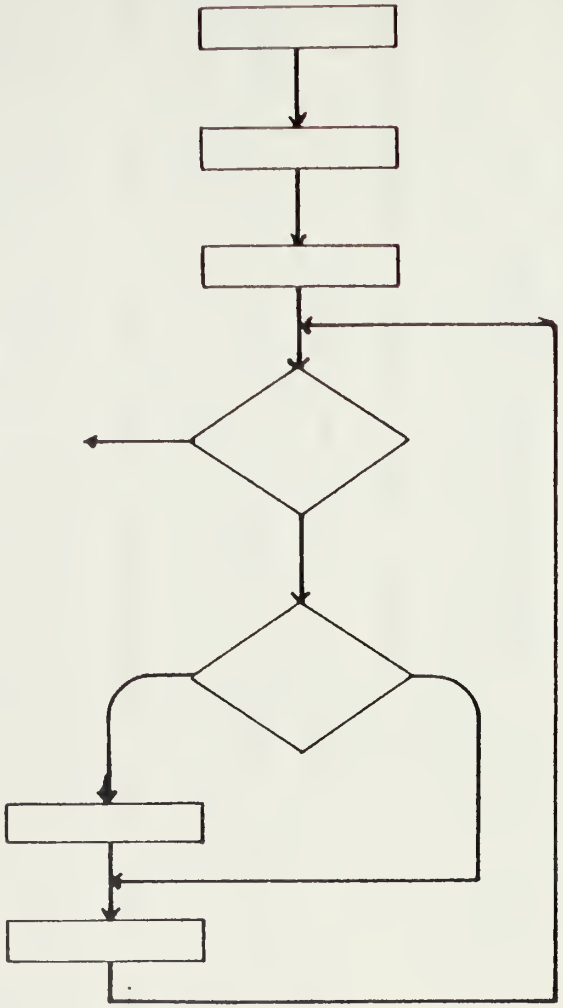
If, for an assignment of values to the input variables x and y, the output variable z will assume a correct value upon execution of the program, then the testing team can assert that the program is correct for this particular test case. If they can test the program for all possible assignments to x and y, then they will be able to determine its correctness. Now, if they assume that x and y are integer variables, and that the program is to be run in a computer with 32-bit registers, single arithmetic, then there are $2^{32} \times 2^{32} = 2^{64}$ possible assignments to the input pair (x, y). Now, suppose that this program in the average takes one milisecond to execute once, then it will take more than 50 billion years to complete the 2^{64} possible combinations of the input pair (x, y). What needs to be pointed out in this example is that no matter how large a practical feasible set of test cases the testing team may choose, it will always consist of an extremely small sample of all the possible cases. This is the basis for the well-known maxim that program testing can be used to discover the presence of errors but not their absence.

A similar problem arises when an attempt is made to exercise all the possible paths in a program. Even though

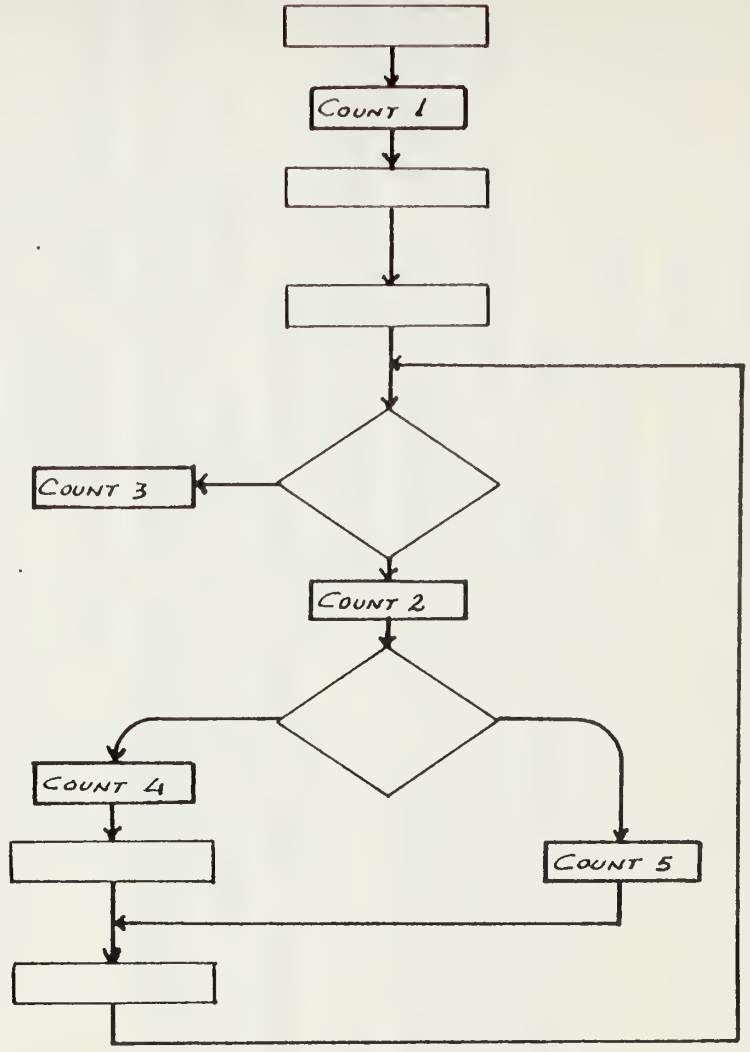
this number is significantly lower than the input combinations, it is still astronomical. However, a practical means is used in order to ensure the minimal acceptable degree of thoroughness of a test. In this process, the flowchart of the program is carefully examined, and some strategic points are determined where software counters can be inserted. Once the test is finished, by looking at the counters, it can be determined which paths were not traversed. Test data cases are prepared for executing those paths at least once during execution. Figure 3.7, is an illustration of a simple program with and without the counters.

The counters can be removed from the program after the testing is completed. However, it is a good practice to leave counters in a program that is used for production, in case more bugs are detected while executing under the production mode, as long as the counters do not degrade performance or introduce other errors. The use of toggles to control the counters, and the use of conditional compilation, make this practice more effective.

The last but not the least consideration refers to the selection of test data cases. Test data cases [32] should be developed for critical input conditions, options, and at the boundaries of all input domains and output ranges. Test data cases should also probe the program's behavior at functional boundaries and for invalid or unexpected inputs. Therefore, testing of a program must consist of the following three phases:



PROGRAM WITHOUT COUNTERS



PROGRAM WITH COUNTERS

FIG. 3.7 EXAMPLE OF THE USE OF COUNTERS

SOURCE: REFERENCE [38]

- Testing the normal cases.

- Testing extremes.

- Testing exceptions.

In all three of them it is common for the program to accept erroneous data and return an incorrect but believable answer. This is something that will normally go undetected with unimaginable consequences.

The idea behind this concept is that programs need to be designed to prevent the use of incorrect data, otherwise considerable precious time will be devoted to searching for bugs that do not exist.

c. Testing Methods

In this section an overview of the testing methods is presented and some examples are given. The methods presented are the most widely accepted in the data processing field. The discussion of the methods is based on proper program testing, and this concept can be extrapolated to subsystem and system testing.

(1) Specifications Testing. It is, and must be, the first part of the complex system testing procedure, and can be divided into two main stages:

(a) System Analyst and the User. During the genesis of the system development process, users and system analysts meet together several times trying to define the user's needs. Interviews, questionaires and other techniques are used to obtain information and define the problem; the product of this process is the so-called Specifications

Manual. Only when system analysts and users agree, can they consider this step finished. Later it may be necessary to go back over the Manual and redefine the requirements.

(b) System Programmer and Analyst. It was established above that the user and the system analyst together write the Specifications Manual. This is the major input that the programmer will have. In case any questions should arise, the programmer should contact the analyst. The following simple rules can be established to assist the programmer in reviewing the specifications and in determining their adequacy:

- The specification manual needs to be reviewed in detail for completeness and accuracy; it should contain a full detailed description of the subsystem and a set of flowcharts, decision tables and/or any other aid indicating all the programs that make up the subsystem.

- If the manual is incomplete, unclear, or not sufficiently specific, the programmer may reject the manual stating in writing the exact reason for this action.

- If the manual is considered sufficiently descriptive, the programmer must accept the specifications.

- Those parts of the manual of major interest to the programmer are:

  - Scope of the manual.
  - Problem definition.
  - General description of the subsystem.

- Flowcharts of the subsystem.
- Decision Tables.
- Input and output layouts.
- Macro-logic where applicable.
- Files or databases to be maintained.
- List of programs and their schedule.
- Controls to be built in the program.
- Glossary.

- After completing the macro-block diagram, the system's analyst will review the logic in detail with the programmer to insure that the programmer understands the requirements specified in the manual.

- All further changes to the program must be approved by the analyst, the programmer, and their supervisors.

(2) Desk Testing. Although this technique is often overlooked, it is necessary at the beginning to be certain that the problem has been clearly stated.

In desk testing the programmer plays the role of the computer; he must follow the logic of the program thoroughly and check to see if it matches the specifications established in the analysis. During this "walking" through the program, the programmer must look for clerical errors, missing cards, missing labels, undeclared variables, and general legibility of symbols.

After checking for possible syntax errors, the most important part of the desk testing technique follows:

the logical flow of the program is verified and tested by using a typical sample case, and again, playing the role of the computer with the help of paper and pencil. It should be pointed out that special care must be taken in verifying the validity of the transfer of control according to all the different alternatives of action that could be present. Also the process involves checking those structures whose behavior depends on well-defined boundaries and the value of index variables, such as iterative loops and data structures like arrays and vectors.

Desk testing does not end with the initial run. After detecting and correcting those bugs that appeared in the initial run, a new pass should be conducted in order to verify the corrections just made and to ensure that no new problems arise because of the changes just made.

After the desk checking is performed the program is submitted to the computer. It should also be pointed out that it is the practice to use the compiler to help find syntax errors; although this is a good idea, it should be exercised judiciously and all syntax errors corrected before using the computer again. No attempt should be made to execute a program without having desk tested it first.

(3) Bottom-up Testing. In this approach the program is merged and tested from the low-level modules up to the high level modules. The bottom-up approach normally requires two stages: module testing and program testing [31].

(a) Module Testing. It is also referred to as "unit" testing. It is considered to be the basic level of testing. This test is conducted in the following manner:

- The modular diagram of the program is analyzed, and all "terminal modules" (i.e., the modules that do not call or refer to other modules) scheduled for the first phase of testing.

- The testing process progresses up to the upper level modules; upper level modules are tested with terminal modules. At this level the interfaces between the various modules of the program and the terminal module are the key testing points. The first modules tested are those that interface directly with the terminal modules. In Figure 3.8, an example helps to demonstrate this concept.

(b) Program Testing. Obviously, at this stage the entire program is tested. The principal types of bugs expected to be detected here are the following:

- Interface errors.

- Complex control and logic errors, especially those due to bad design of the subsystem or poor usage of the modules.

Now the concept can be expanded to subsystem and system testing.

In the framework of this discussion a subsystem has been described as a collection of programs. In order to test the subsystem, the module testing approach is used. The difference is that now the modules are programs, but the philosophy is the same.

BOTTOM-UP TESTING EXAMPLE: The following is a listing of a fictitious program that consists of ten (10) modules. The relationship between these modules is represented by the CALL statement.

```
Main Program
...
Call B
...
Call C
...
End of Main Program
```

```
Subprogram B
...
Call D
...
Call E
...
End of Subprogram B
```

```
Subprogram C
...
...
Call I
...
...
End of Subprogram C
```

```
Subprogram D
...
Call E
...
Call J
...
End of Subprogram D
```

```
Subprogram E
...
Call F
Call G
Call J
...
End of Subprogram E
```

```
Subprogram F
...
Call G
...
Call H
...
End of Subprogram F
```

```
Subprogram G
...
Call H
...
End of Subprogram G
```

```
Subprogram H
...
...
...
End of Subprogram H
```

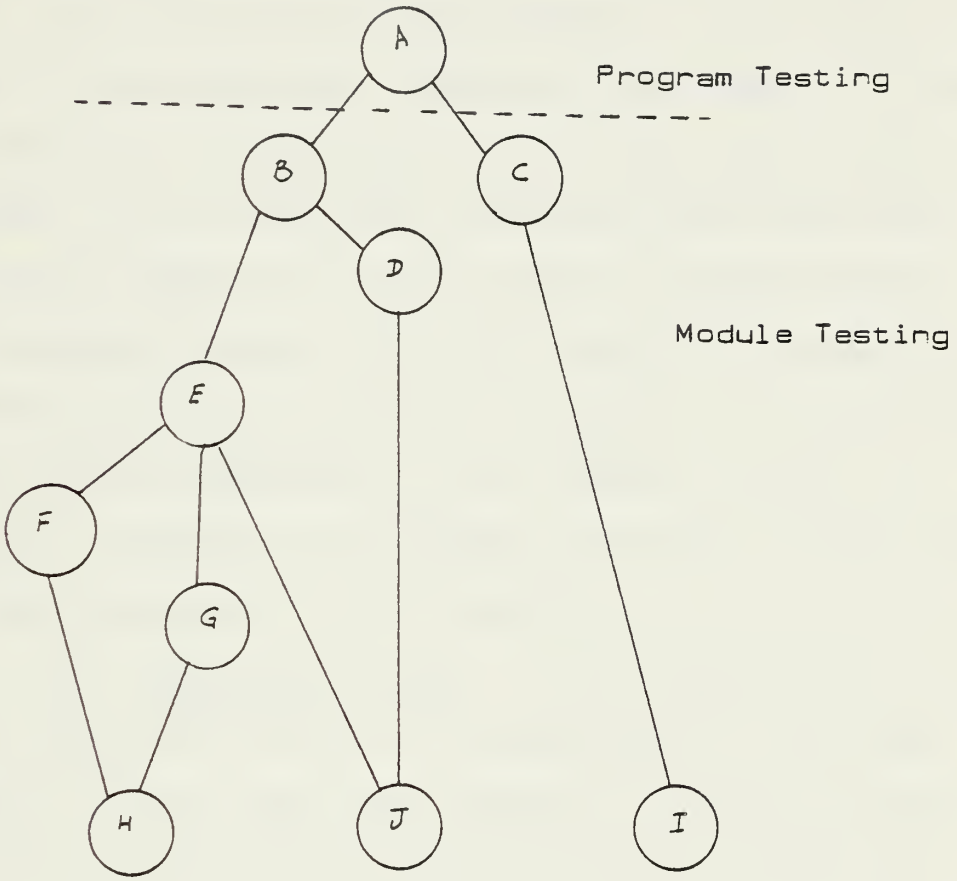
```
Subprogram G
...
End of Subprogram I
```

```
Subprogram J
...
End of Subprogram J
```

Fig. 3.8 Example of Bottom-up Testing

Source: Reference [31]

A tree like representation of the modules to be tested follows:



Module Testing: a) Test terminal modules H,J and I
b) Test upper-level modules in the following sequence:
- Test G with H, and C with I
- Test F with G, and H
- Test E with F,G,H and J
- Test D with E,F,G,H and J
- Test B with D,E,F,G,H and J

Program Testing: To test module A (Main Program) with the rest of the modules.

FIG. 3.8 (CONT.). EXAMPLE OF BOTTOM-UP TESTING

SOURCE: REFERENCE [31]

A system has been described in this discussion as a collection of subsystems; therefore, the above discussion is applied except that modules are now the subsystems. The question of when testing will take place is a problem with this appraoch. Obviously, the answer is when all the subsystems are developed. But it may take several years to have all the subsystems developed for a particular system, and the reader will recall that the system development is a dynamic process. The ideal situation of having all the subsystems developed may never be reached. Other factors that cause system development to be a dynamic process are improvements in hardware and software technology. The writer does not have an answer to this question.

(4) Top-down Testing. In this approach [31, 37], the method consists of tests of the program from the high-level modules to the lower-level modules. Top-down testing suggests that testing should provide the main program and perhaps one or two levels of subroutines as a "skeleton" which is separately tested. The high-level modules can call lower-level modules that may not exist concurrently. During the test it will be necessary then, to implement low-level modules as dummy modules to simulate the functions of the missing modules.

Top-down testing is not considered a rigid approach, but is intended as a philosophy, and it can and should be altered as the situation requires. It is sometimes quite difficult to implement dummy modules to replace all

routines that are called by high-level modules; obviously, the content of the dummy modules will depend on the type of function desired to be implemented, especially when the module being called must provide some type of output parameters that are needed by the calling module. It is with this concept in mind that the testing team can design dummy modules (also called stub modules) that will return control as soon as they have assumed it. The simple ones only contain the executable statement RETURN; others contain enough statements to generate random parameters or constant output. In the case of utilizing random output, care should be taken in order to avoid the generation of data that lies out of range. As an example of this, consider a program written to solve the following quadratic equation:

$$Ax^2 + Bx + C = 0$$

It is known that a valid value for A is one that is not equal to 0 (zero), because otherwise, an error of division by 0 (zero) will occur; if the testing team is using some type of pseudo-random number generator, they must ensure that the value of 0 (zero) is never returned in order to avoid this problem.

Once the "skeleton" has been tested, the team can proceed and add one new module at a time until the whole program has been tested. Some of the advantages of this approach follow:

- Due to the fact that only one piece of code is added at a time, if something goes wrong, the source

and nature of the bug will be usually more easy to identify.

- Most major interface, control and logic bugs are discuvered at an early stage in the testing process.

- At a relatively early stage the basic logic of the program is working.

Figure 3.9, illustrates the concepts just presented in a graphical way.

(5) Big-Bang Testing. The philosophy behind this method [37] establishes that all the modules should be tested individually and isolated from the others. After testing all the modules alone, they must be integrated together as a whole for program testing. Obvious disadvantages when using this method follow:

- For each one of the modules to be tested dummy modules and module drivers must be constructed.

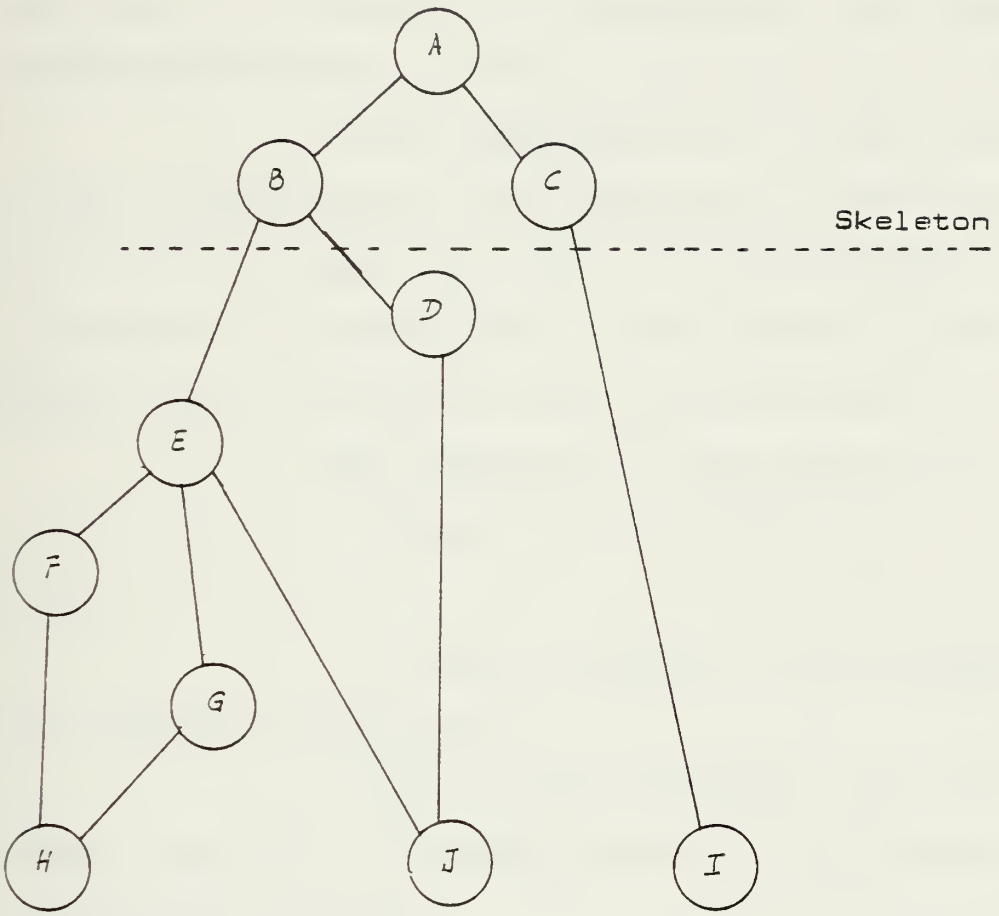
- Integration of modules occurs too late in the process. This will probably cause a delayed detection of interface bugs.

- Debugging is more difficult, since the testing team is testing all the modules together for the first time, and errors could occur in almost any part of the system.

- It is possible that a working program will not be available during the early stages of the process.

This method is not recommended for the testing of large programs, and one of the things that needs to be taken into consideration is how the team/individual will

TOP-DOWN TESTING EXAMPLE: Using the same tree-representation from Fig. 3.8



Everything below the line is initially simulated by a dummy module. Modules D,E,F,G,H,I and J will be added one at the time to the skeleton.

Fig. 3.9 Example of Top-Down Testing

Source: Reference [31]

psychologically react after spending long periods of time testing the system wihtout achieving tangible results.

(6) Sandwich Testing. This method [37] combines the main features of two techniques already presented: Top-down and Bottom-up, trying to overcome their disadvantages.

In this technique both Top-down and Bottom-up begin simultaneously; the integration of the system is made from both the top and the bottom and eventually meet somewhere in the middle. Deciding where to meet depends on the tester's judgement and the system's design and structure.

Some advantages of the method are:

- Integration is obtained early in the process.

- A working program will be obtained in the early stages of the process.

- The problem of having to construct stub modules tends to be diminished because of the presence of the Bottom-up technique.

Some disadvantages are:

- It is still necessary to write driver modules and stub modules; sometimes these will be complicated programs, depending on the complexity of the relationships internal to the program.

- It is difficult to determine where to stop the different approaches and to start testing the program as a unit.

(7) Other Testing Techniques. There are a variety of other testing techniques in the field of program testing, but most of them are still in an experimental stage and require further research. A very brief description of some of them follows:

(a) Redundant Programming. The key point behind this idea is to have two or more independently designed and written programs for a particular application. Each program will be designed, coded and tested by different teams; the only thing in common to all of them will be the specifications for the application. An obvious disadvantage for this method is the amount of additional programming and system overhead required [31].

(b) Standardization. In this approach, the tendency is to reduce the occurrence or presence of errors by using general purpose subroutines and packages for designing programs. Even the subroutine packages have hidden bugs, but their behavior may be predictable and known. This is better than dealing with a special-purpose program that may contain bugs [31].

(c) Simulation. In the development of large and complex systems it is common to use simulation, which can be very useful for evaluating performance. In a few cases it can also help to discover logical errors, but it certainly will not help to find coding errors.

Schneidewind, in reference [39] shows how simulation can be used to evaluate alternatives during design and to simulate the detection of errors during testing.

(d) Mathematical Models of Program Reliability. The requirements of software reliability in the more complex systems, have recently led to a great deal of research in the development of mathematical models for program reliability. By gathering statistics during the testing process, the model attempts to predict, in a probabilistic sense:

- The number of bugs remaining in the program when it is put into production.

- The mean time between failures of software.

- The probability that the program will run successfully for a specified period of time.

This method has been used rather successfully in space projects and in some defense projects, but there is still some hope that it could be applied to commercial systems in a few years [31].

d. The Testing Plan.

The starting point for testing was accomplished during detailed design, where the preliminary test specifications were written; however, a more detailed plan is required at this time for those responsible for conducting the test (the testing team). The elements of a good testing plan are [37]:

(1) Objectives. A definition of the goals of each test phase: module, program, subsystem and system testing.

(2) Responsibilities and Schedules. A definition of who will perform test case design, development, execution, and debugging for each test phase and information concerning the place, and the data when testing will be conducted.

(3) Tools and Methods. A description of the needed test tools, and the testing methods that will be used, for each phse.

(4) Computer Time. An estimate of the computer time required for each phase.

(5) Configurations. A description of any special hardware or software configuration needed to conduct the test.

(6) Test Case Libraries and Standards. A definition of how test cases will be stored and standards for writing test cases.

(7) Tracking Procedures. A description of the method to be used to monitor test progress.

(8) Debugging Process. A definition of the procedures needed to report and correct errors.

(9) Acceptance Criteria. The criteria that will be used to judge successful completion of each test phase. The establishment of these criteria is the user's responsibility.

e. Acceptance Testing

Acceptance testing is a validation process in the sense that it tests how the subsystem matches the user requirements. This test must be conducted by the user organization

who designs and writes the test data cases and tries to make the subsystem fail. If the subsystem does not fail and meets the minimum requirements, it can be accepted for use [37].

f. Manager and Testing Team Responsibilities During the Testing Process [17].

Manager's duties.

- Establish acceptance criteria.
- Write and design test data cases for acceptance testing.
- Conduct acceptance testing.
- Accept Subsystem.

Testing Team's duties.

- Establish testing plan.
- Assist user organization in preparing and conducting the acceptance test as needed.
- Record and generate necessary documentation during testing process.

g. Documents Generated During the Test Process

There is a set of documents that serve as input to this phase, and they have been generated in previous steps of the system development. In general, they cover the following:

- Subsystem design documentation.
- Subsystem and program design documentation.
- Debugged programs.

To the above documents the following are added:

- The detailed test plan - already discussed.

- All the test data cases produced, whose secondary purpose is to act as historical documentation for future reference.

- A report containing the following information: characteristics of the errors found, location, nature of the error, how it was corrected, and information about the testing process.

2. System Implementation

System implementation deals with the concept of bringing a developed system into operational use and turning it over to the user.

The implementation of a computer system affects both the hardware utilization and the personnel that will use it.

Putting a new system into operation is usually complicated by the fact that there is an older system already in operation. The implementation team has to deal with changing from something familiar to something new and different. Some of the main activities that make up system implementation are the following: Training of personnel, the conversion of programs and files, the installation and checkout of the new equipment, and the beginning of new operations [21].

A description of these activities follows below:

a. User Training

The idea, here, is to train all those people who will be part of the day-to-day operation of the system to be implemented, because it is vital for making the system work. Some of the most important points that should be covered in a training plan of this nature are the following:

- General information about the system.

- Information about specific operations of the system.

- Give to the user some "hands-on" practice in operating the system.

- Get some feed-back from the user.

The last two points are particularly important to the successful implementation of the system, because through practice the user will become more familiar with the system, will gain more confidence on its operation, will pinpoint potential problems areas that were not foreseen during system design. By using this approach the user becomes a participant in the implementation process [2].

b. Preparing for New Equipment

This concept deals with all the activities necessary for the installation of the new equipment. This activity started after the selection and evaluation of the computer system phase in order to meet the delivery schedules of the supplier or to build a new computer facility if needed. This activity consists of the following steps: site preparation, installations of data processing equipment, and hardware and software check out [21].

(1) Site Preparation. All the requirements necessary for site preparation were established when the computer system was condigured, according to the information that was supplied by the manufacturers.

The provision of adequate site preparation storage cabinets, magnetic tapes, cards and other supplies,

is the responsibility of the data processing department. Basically, provisions are established for the following:

- Adequate work space for the personnel, the hardware, office material, and the like.
- Power requirements, established in terms of: voltage and tolerance, frequency and tolerance, number of phases, and power consumption.
- Illumination of data processing center.
- Floor.
- Ceiling.
- Air conditioning.
- Safety controls, like smoke detection equipment, fire extinguishers, and the like.

(2) Installation of Data Processing Equipment. Once the site has been prepared, the installation of the new equipment can take place. The installation is made by manufacturer's personnel with the help of organization's personnel.

(3) Hardware and Software Checkout. The hardware and software systems are normally tested by the manufacturer's personnel. Usually this checkout will include the running of special test programs provided by the manufacturers.

All the hardware system, compilers, assemblers, operating system, utilities, data communication equipment, etc., must be thoroughly tested before being delivered to the organization. The data processing department must be involved in this phase with the manufacturer.

A demonstration of the system with some test applications is necessary before delivery.

c. Programs and Files Conversion

The type of data conversion can vary according to several parameters. They are the following [3]:

- Conversion of data that presently exist on machine readable media.

- Conversion of data, presently manual, which must be converted into machine readable format prior to conversion.

- Conversion of data that does not contain all the information required for the new files.

- Conversion of files that exist as separate files into an integrated data base.

- The size of the file being converted.

Since most systems will have multiple files to be converted into one or more new files, the order in which the files are to be converted must be considered.

d. Beginning of New Operations

This concept deals with the problem of transition from the old system to the new one. Basically three concepts are discussed: Immediate cutover, phased cutover, and parallel processing. The description of this process as quoted in reference [3] follows:

(1) Immediate Cutover. The idea is to stop operations under the old systems and begin immediately with the new system. The disadvantages that this approach has are the following:

- It will shock the organization - no adjustment period.

- In the event of failure the whole process must be repeated with dual inconvenience of returning to the old system and then implementing the new system again.

- It requires a very brief period for changeover with consequent strain on the organization.

- It requires scheduling for minimum workload period when all required personnel are available.

The advantages of selecting this approach follow:

- The new system is immediately available and can be exploited fully.

- If no major problems occur, the cost savings can be very significant.

(2) Phased Cutover. The new system is implemented by phases. This means that part of the operations are carried out by the old system and part for the new system. The disadvantage of this method follows:

- The scheduling of cutover of each phase or subsystem becomes very critical and complex. For each subsystem there is an alternative of immediate cutover or parallel processing.

- The organization of the user departments and/or the nature of the system may be such that it is not possible to implement the system by its subsystems.

The advantages of this approach are the following:

- The outputs of the implemented subsystems are utilized prior to full system implementation.
- Failure of a subsystem is not as critical and recovery does not involve the whole system.
- User experience with a well designed subsystem will make the acceptance of the remainder of the system easier.

(3) Parallel Processing. Both systems, the old and the new one, are maintained in operation simultaneously for some period of time. Both systems will be kept in this condition as long as necessary in order to assure a highly reliable new system. The disadvantages of this approach are the folloiwng:

- The direct costs can be very high as almost complete duplication of processing will be required.
- The new and old organizational structures may be incompatible, making parallel processing infeasible.
- The volume of work generated by two systems may be too large to handle.

The advantages of this mode of operation follow:

- Implementation will have a system that has been fully checked out under actual operating conditions.
- Failure of the new system will be less severe, as normal processing can continue to handle the work.

- User personnel will have the time to become completely familiar with the new system.

- The outputs of both, old and new systems, will be available for comparison and evaluation.

3. System Operation

In this phase the control of the system is passed from the project team to the operations group. Now, they have the total responsibility for the operations of the system.

In this phase the system is tested in a production environment. From the viewpoint of testing, it is a validation process, because how the system behaves in a "real world" is the main consideration at this point.

Although the responsibilities for the system have been shifted from the designers to the operations group, an open communication channel must exist between both groups in order to solve all those problems that may arise after implementing the system. In general, these problems can be grouped in the following areas [3]:

- Previously undiscovered errors that become apparent when the system is in operation.

- Changes in the system software that require changes in the application system.

- Changes in system hardware that require changes in the application system.

- The implementation of new application systems that require modifications to existing systems because of interface requirements.

- The growth of applications systems that require changes or redesign of existing systems.

- The familiarization of the user with the system will allow him to discover areas that may be improved, and that will require the modification of the application systems.

- The above concept is also applicable to the operations group who can find areas of the system that can be improved or changed in actual operational procedures.

- Changes in external legal requirements may cause modifications in application systems.

Another consideration is operational procedures which can be grouped in two main areas:

The first, affects the operations management and the second affects the project team. Operations management is not treated in this discussion because it is not a part of the system life cycle. The operations topics that are of interest to and the responsibility of the project team follow [4]:

a. System Conformity to Installation Operations Standards

Normally in a data processing organization, procedures are established to govern the computer center operations. Systems should be designed in such a way that the established standards are not violated. Violations may be detected during the operation of the system, and they must be reported to the system group for corrective action.

b. Usefulness of Operation Manual

Operator's manuals were developed during the design phase. They are the communication link between the

operator and the application systems. This means that as the operator becomes more familiar with the system, he may find better ways of processing it than the methods established in the manuals. Therefore, the manuals are revised in order to improve the operator's productivity and system efficiency.

c. Evaluation of Run Efficiency

During the system design phase the project team is working under constant pressure created by user demands, and tight schedules. The designers devote their efforts toward attaining a successfully running system; once the system is running well, they try to improve its efficiency.

Normally, improvements can be performed in areas such as a better use of the hardware and software capabilities, file handling, program coding, timing, and throughput.

During system operations all the modifications or changes must be asked for in a formal document. This formal document will be a part of the historical file of the applications, and will also be a supporting document for scheduling the modifications and for keeping track of all the improvements that have been performed during the life of the system. This document is known as the request for changes or modifications and it will be explained in the next section.

4. System Maintenance and Follow-up

System maintenance is an activity through which changes or modifications in a system are performed in order to meet the established requirements of the system.

This activity begun in the system operations phase, where the system has been under continuous observation by the

operation group. In that phase was established the necessity of recording in an appropriate document all the problems observed by the operations group or the user. These documents are the input to the maintenance and follow-up activity.

The documents generated in the system operations phase are analyzed by a maintenance team; this analysis will allow them to establish a set of actions that can be carried out to maintain the system. These actions could be the following:

- No changes.
- Improve the system.
- Replace the system with a new one.
- Discard the system.

The last three actions must be carefully controlled because they may affect other systems in the MIS environment. Modifications, although minor, could notably affect the objectives and design considerations of a MIS.

It can also be noted that if replacement is the chosen action, the system will come under the next step of the development process, system cessation, and the whole cycle starts all over again for a particular system.

The most critical situation during system maintenance is control. Control in this sense covers two points: control of the changes that are necessary and control to prevent unnecessary changes.

Changes or modifications that seem to be easy, from the user's viewpoint, may cause a significant time consumption

for analyst, programmers, operators and the computer itself. Also possible side effects may be caused in other systems.

There are times when the modifications requested by the user are insignificant in the sense that they do not improve system efficiency, or do not improve operator's productivity or do not improve user's work performance.

One way of dealing with this situation is through the request for change or modification.

Basically the report contains the following:

- Date of request.
- User's department identification.
- System identification.
- Change(s) or modification(s) requested.
- Benefits that will be obtained.
- Costs. This calculation must be performed by the data processing department.
- Signature of the authority that approves the request, preferably at middle/top management level.

Since the report must be signed by an authority of the user's department, chances are a more careful study concerning the need for change will be conducted.

System maintenance and follow-up is a necessary activity in the system development process.

The roles of the user and the system analyst during this phase follows:

a. User and System Analyst Responsibilities During the System Maintenance and Follow-up [17]

Maintenance and follow-up must be undertaken if the system is to be successful. Since systems are dynamics they require monitoring in order to meet the needs of the changing environment in which they function.

- Manager's Duties
  - Notes changes in environment that affect the system.
  - Makes the request for changes or modifications.
- System Analyst's Duties
  - Audits the system.
  - Schedules changes or modifications according to the user's request.

Overall: Continuous monitoring of the system and its environment by the manager and his staff and the system analyst.

5. System Cessation

This is the last stage in the system life cycle. Systems will be used for some period of time and after that they will be discarded. The useful life of the system will be dictated by the environment in which it is being used. The two main factors that will determine the end of a system are the following:

- Substitution of a system by an improved system.
- Discontinuance of a system when it is no longer useful to the organization.
- Economic and technical factors.

The continuous surveys made during the system operations, maintenance and follow-up stages are very useful in

providing some mechanism for determining when a system has reached its cessation point. For example, when numerous modifications and patches have been made to a system, or if new modifications are going to be extensive, the practicality of designing a new system should be considered.

The question arises regarding the MIS cessation point. This question will persist because MIS will always be needed in an organization because information is necessary for the continued operation of the enterprise, whether it be manual or computer based.

IV. CONCLUSION

A study of the system development process has been presented. This study emphasized the system life cycle, which covers the areas of system analysis, system design, and system implementation.

The starting point of this discussion was the conception of a Management Information System that could satisfy the information requirements of an organization.

Information is a vital ingredient for management. Information is necessary to formulate objectives and policy which subsequently must be interpreted and communicated to many people. The evaluation of these objectives and the subsequent decisions and action plans are based upon information concerning their impact on the organization. If this information is incomplete or inaccurate, it can limit the efficiency of management. The objective of a MIS is to make available comprehensive and accurate information, and to provide a systematic method of controlling and directing this vital management resource.

Management information systems do not happen; they have to be uniquely constructed to fit the organization they are to serve. The MIS development generally follows a master plan; this master plan contains three major phases: MIS Analysis, MIS Design, and MIS Implementation. This thesis described the development of the master plan.

The support and participation of the user and the data processing group was established as one of the most important factors for a successful MIS development.

The design of a large integrated MIS must provide operating efficiency and the ability to adapt to inevitable changes of the organizational environment within which it must survive. Several design considerations are the key to success in both areas. Such factors as the analysis of information flow, modular design, structured programming, top-down testing, the implementation of databases - all interact to establish the operating characteristics of the system. Each of these areas were treated throughout this discussion. The future trends in MIS are oriented toward the implementation of distributed systems.

APPENDIX A

I. THE SYSTEM DEVELOPMENT TEAM

The organization of the system development team is another of the key points in an MIS success.

Obviously, the purpose of the system development team is to carry out the development of a MIS to its final stages. Today, most organizations have their own policies for recruiting and evaluating personnel. It will be assumed in this discussion that through those policies the organization is obtaining skilled and qualified personnel that meet the standards of quality and knowledge desired by the company.

Composition of the team involves both representatives of the user's organization and the data processing department. Prior to exaplining how the team should be organized, a description of the main functions that are carried out by the team is given.

A. SYSTEM DEVELOPMENT TEAM FUNCTIONS

There is almost a one-to-one correspondence between the functions performed by the team and the phases of the system development process. These functions are:

1. Analysis and Design Function

This function deals with the gathering of information from the early stages of development, user's interviews, analysis of the actual system, design of a proposed system,

implementation strategies, and hardware and software specifications.

2. Programming Functions

Although this function is normally included in the design phase, programming is treated separately in this context. The purpose is to translate the design specifications of the system into executable programs that will run in the computer system. The use of modular and structured programming techniques, top-down, bottom-up, or other design methods, codification and debugging, are the main activities carried out in this function.

3. Testing Function

The test of the systems, in order to demonstrate that they do what they are supposed to do, is the key activity in this function. Selection of the testing methodologies test data cases, planning of the testing efforts and other related activities are the main functions performed.

4. Implementation Function

Once systems are tested, they are put into production. The user and the operations group will now have control of the system. File and data conversion, operations procedures, maintenance and follow-up are the characteristic activities performed in this function.

5. Data Management Function

The data management function is exercised through the Data Base Administrator. The activities performed here are related to the database (DB) and they are the following:

establishment of the data base maintenance, growth control, updating, error detection and correction, information retrieval, security and recovery.

6. Documentation Function

This function deals with the administrative activity of the system development process. The preparation of all the necessary documents that support the system development effort is the main goal of this function. Documents in this activity range from the most insignificant one page report to the most sophisticated user's manual, operator's manual, or system design manual.

7. Hardware/Software Evaluation Function

The establishment of hardware and software specifications necessary to make the implementation of the application systems possible, the translation of those specifications into a computer system configuration, and the evaluation of the computer system are the main activities performed by this function. This activity involves the establishment of requirements and evaluation of processors, memory, mass storage device, data communication equipment, operator's console, terminals, printer, operating system, programming language, and software packages.

8. Human Engineering Functions

Human Engineering or human factors as it is sometimes called, combines the scientific areas of computer technology, psychology, methods analysis, and industrial engineering. Its objective is the optimization of the man-machine interface

in an MIS. The activities performed within this function deal with the design or selection of terminal equipment, selection of the language most appropriate for communication with the system, definition of the role of graphics for output display, and determination of the appropriateness of on-line and off-line procedures [12].

9. Training Function

Training involves a set of activities necessary for good operation of the system. The training is normally oriented to the user's organization and the operations group. Through training, users will know how to collect data, fill out inputs forms, and other manual procedures, and interpret the output results. The operators will learn how to: organize the input data for the computer, identify (label) files, handle error situation, and carry out recovery procedures.

10. Users Advisory Function

This activity is exercised by the user's organization. The idea is to assign qualified personnel within the user's organization that can act as a consultant-for the data processing group in matters concerning the application systems under development.

11. Planning Function

Preparation of schedules, time development estimates, and establishment of priorities, are some of the typical activities involved in planning.

B. SYSTEM DEVELOPMENT TEAM PERSONNEL REQUIREMENTS

The functions explained in the above section are accomplished by personnel who must possess the skills listed below.

1. System Analyst

He is a data processing specialist. Some of the desired skills of a system analyst are: at least an undergraduate educational background, a logical mind, an ability to work with others, an ability to solve problems, creativity, initiative, and communication ability. He must be familiar with design techniques, programming languages, flowcharting and decision tables techniques, hardware devices, software packages and testing techniques.

2. Programmer

He is a data processing specialist. He must be familiar with program design techniques, modular and structural programming, proficient in programming languages, and knowledgeable in hardware and software capabilities, and testing techniques.

3. Database Administrator

The database administrator (DBA) is not necessarily a data processing specialist, but it is desired that he have technical and management backgrounds. He must be familiar with the database packages, languages, operating system characteristics related to the database, and the hardware requirements for supporting the database.

4. Maintenance Engineer

He is normally skilled in equipment diagnosis with knowledge of peripheral devices, transmission lines, modems,

terminals, memories and other related computer equipment.

5. Software Engineer

He is a data processing specialist. He is familiar in detail with operating systems, languages programming, software maintenance and other software areas.

6. Technical Writer

He is not necessarily a data processing specialist; although, some training in this area is desired. He must be able to translate the technical information generated through the system development process into an understandable document. He must also be able to write all those documents mainly oriented toward the users, as for example the user's manual, in a non-technical language.

7. Human Engineer

The human engineer specialist must possess characteristics that are very difficult to find in one person. Ideally a human engineer specialist should have several degrees ranging from Ph.D's to Bachelor's degree in areas such as: experimental psychology, electronics, mathematics, computer technology, anthropology, physics and pedagogy, dealing with machine and human behavior. In order to overcome this problem the human engineering function is exercised by a team of specialists with the above backgrounds [40].

8. User Advisor

He is normally a professional in the user's organization, capable of assisting the data processing group in those areas concerning the user's application system.

C. STRUCTURE OF THE SYSTEM DEVELOPMENT TEAM

In this section the structure of the system development team is explained. The discussion that follows is general, since each organization is free to make their own adaptations according to their particular situation.

The system development team for a MIS should be composed of two groups. One group is the system development control group and the other group is the project development team. The composition and functions of each group follows:

1. System Development Control Group (SDCG)

The SDCG is the group that is charged with assuring the success of the MIS development. Basically, this group deals with: interface problems between the major functions of the MIS, planning activities, allocation of resources, organizational changes, and organizational conflicts. The composition of this group may be as follows:

a. The MIS Director

He should be selected by the organization; preferably an executive at the top level who will coordinate all the activities of the group. Some organizations choose for this position a manager from the user area. Others like to designate a data processing specialist. Of his responsibilities the following are the most important [12]:

- He must translate the requirements of top management to the MIS staff.

- He must communicate the constraints and capabilities of technology from the MIS staff to top management.

- He must direct the MIS development effort by establishing budgetary and schedule goals and must measure progress against goals.

- He must be able to challenge the MIS staff's demands for additional resources, when they are not justified, and to support these demands to top management, when they are justified.

b. The MIS Director's Assistants

Since the MIS director may be a non-technical professional, he needs colaborators who are technical experts. These experts are the following: senior analyst/programmers, system planners, specialists from the user's area, database administrator, and clerical personnel. There should be as few as are necessary for carrying out the required functions.

2. The Project Development Team

The project development team is in charge of the design of each subsystem that is a component of a MIS. There will be as many project development teams as there are subsystems under development. The project development team reports directly to the system development team control group. The approach used in the composition of this team is the one called the chief programmer team. This team may be composed of the following way [41]:

- A chief programmer who is in charge of the activities of analysis, design, programming, testing, and documentation of the subsystem.

- A back-up programmer who must be able to substitute for the chief programmer at any time. His main function is to help in the design as a thinker, discussant and evaluator.

- An administrator who controls the resources allocated to the project: money, personnel, space, machines and materials.

- A technical writer who is responsible for translating all the manuscripts produced by the chief programmer into readable and understandable documents.

- Two secretaries, one for the administrator and the other for the technical writer. They handle project correspondence and non-product files.

- A program clerk who is responsible for maintaining all the technical records of the team in a programming-product library.

- The tester, who is responsible for preparing all the necessary test data cases and the test plan.

3. Other Items

In many organizations the project effort, from the initial study until the maintenance and follow-up phase, is the responsibility of the project team. This includes hardware/software configuration and evaluation. This can be carried out by a team composed of some chief programmers and backup programmers. However, other organizations may have other teams in addition to the project development team. These teams perform some of the functions described at the beginning of this Appendix. Basically the structure of the

project development team which includes other teams is the following:

- Project development team previously explained.

- Evaluation team who will perform the functions of configuring and selecting the computer system.

- Testing team who will perform the activities of the testers in the project team approach.

- Implementation team who will perform the functions of installation, operation, maintenance and follow-up. This team should be considered by any data processing organization because the demand for system maintenance can be such that the project team will have to devote its effort to this activity rather than to developing systems.

- User advisory board. This is a group of qualified professionals of the user's organization who will assist the project team in all those areas concerned with the subsystem under development.

- Human engineering team which will perform the functions that were explained in previous sections.

The main disadvantage of creating many teams is the manpower requirement and the activity of control is required in order to obtain work coordination. However, the approach of having several teams can speed up the development process by creating a partition of work in a large system.

APPENDIX B

I. THE DATABASE APPROACH

During recent years data base systems have received greater attention from computer specialists. The terms data base, data bank, corporate data bank, generalized data base, and common data base are used synonymously to indicate the central repository, in a logical sense, of all automated data available to the organization. The concept of a data base system extends the definition to include the capture, update, storage, manipulation, and dissemination of data used within the organization in a controlled, consistent fashion.

The general objective of a data base approach is the elimination or minimization of data fragmentation, data redundancy, data inconsistency, and inconsistent data manipulation which is typical of many file processing environments [43, 44].

A. PLANNING FOR THE DATABASE SYSTEM

The commitment to a data base approach is a major organizational decision. Essential to the success of the implementation of the data base system is the support of the user, management, and data processing groups. The main components of a database plan are the following [43, 44, 45, 46]:

1. Definition of Goals

The data base plan should be a subset of the overall data processing plan. Development of the database system is based on a general statement of goals to be achieved within the data base system; as such, these goals must based on the organization's long range plan. These objectives must then be communicated to the user, management, and data processing groups. They must include the estimates of costs, resources, time required to develop a database system, and the responsibilities of users, management, and data processing.

2. Establishment of Data Base Administration (DBA) Function

The development of a data base-supported MIS requires the standardization and coordination of definition, capture, storage, update, manipulation and dissemination of data. These requirements are administered and centralized in a function defined as the Data Base Administration function, with the Data Base Administrator being responsible not only for the coordination and control of the data base, but also for the establishment of the standards and procedures for coordination and control, and the evaluation and selection of the software required.

The Data Base Administrator and his staff comprise the data base development team, which is in charge of the execution of the data base plan.

3. Development of Data Collection Standards

The development of data collection standards is a key task in the data base plan. The data collection standards

lead to the identification and documentation of the existing and projected information needs of the organization.

4. Acquisition and Installation of the Data Dictionary/Directory System

The Data Dictionary/Directory System (DD/DS) is one of the most important tools for the coordination and control of the data base system. The DD/DS interfaces with the data administrator, the system analyst, the programmer, the user, and the various software elements of this component of the data base system. The dictionary function answers the question, "What data is available or contained in the organization's data base?" The directory function replies to: "Where is the data stored?"

5. Documentation of the Present and Project Data System

This process is an extension of the data collection procedures. The Data Base Administrator records the collected data as it currently exists and as it is expected to be.

6. Analysis of Documented Data

The data documented in the preceding step is carefully analyzed for common areas of information, as well as data redundancy, data inconsistency, and application interrelationships.

7. Selection of the Nucleus System

The analysis of the documented data, performed in the prior step, forms the basis for selecting a nucleus application area that represents the first system development effort under the integrated data base system. The purposes of identifying the nucleus are:

- Definition of the initial applications to be implemented in the data base system.

- Establishment of requirements for selection of a data base software package.

8. Definition of the Logical Data Base Structure

The Data Base Administrator describes all pieces of data and how each relates, according to its many uses across the organization. The logical design is totally independent of any particular data base software package.

9. Evaluation and Selection of a Data Base Software Package

The Data Base Administrator establishes the performance criteria against which the data base software package is measured. In addition to the proposal of the various data base software vendors, the Data Base Administrator should take into consideration the alternative of in-house development. In evaluating and selecting the data base software, special attention must be placed on how it interfaces with the operating system, language compilers, and hardware available in the organization.

Mathematical modeling, benchmarking, and simulation methods are normally used to evaluate data base performance.

10. Integrate and Install the Data Base Software

The final task of the data base planning is the integration, testing, and acceptance of the data base software. The Data Base Administrator is responsible for establishing the testing plan and acceptance criteria. Upon completion of

this task, the data base system is implemented, and the Data Base Administrator will coordinate the system and data base design functions according to the standards and procedures established by the Data Base Administration function.

B. DATA BASE DEVELOPMENT AND THE SYSTEM DEVELOPMENT PROCESS

Several aspects of the traditional system development cycle are not affected; however, there are crucial areas that must be modified as an organization moves to implement a data base supported MIS. The modifications are the following [47]:

- Separation of the data definition and data base design function from the system design function.

- Project Development Team is responsible for analysis of user requirements, detailed specifications, coding, documentation, testing and implementation.

- Project Development Team is responsible for the identification of data requirements.

- Project Development Team is not responsible for file design. It will submit a request for data to a data base design group, a group that reports directly to the DBA.

The separation of the data base design function and system design function is necessary in order to assure that the standardization and coordination of data definition, capture, storage, update, manipulation and dissemination are observed; however, a close communication must exist between the data base design group and the project development team throughout the system development process to ensure complete understanding and agreement as to data characteristics.

C. DATA BASE DOCUMENTATION

The same principles that applied to the system documentation phase can apply to the data base documentation phase. This important responsibility includes the recording of procedures, standards, guidelines and data base description necessary for the proper, efficient and continuing utilization of the data base environment. Reference [48], outlines the following documents as some of the most important:

- Description of Data sources, where the Data Dictionary is the primary initial tool for determining potential sources of information.
- Data base detailed specifications.
- Description of the data base which includes logical and physical data base organization and data attributes.
- Standards, which include the standards for data collection, capture, storage, manipulation and dissemination.
- Data access and manipulation procedures.
- Passwords and user identification.
- Back-up procedures, which include identification of the data to be backed-up, back-up facilities to be used, and back-up schedule.
- Restart and recovery procedures.
- Data base testing sepcifications.
- Data base training, education procedures, and manuals.

LIST OF REFERENCES

1. FitzGerald, J. M. and FitzGerald, A. F., FUNDAMENTALS OF SYSTEMS ANALYSIS, John Wiley and Sons, 1973.

2. Semprevivo, P. C., SYSTEM ANALYSIS: DEFINITIONS, PROCESS, AND DESIGN, Science Research Associates Inc., 1976.

3. Hice, G. F., Turner, W. S., and Cashwell, L. F., SYSTEM DEVELOPMENT METHODOLOGY, North-Holland/American Elsevier, 1974.

4. Rubin, Martin L., INTRODUCTION TO THE SYSTEM LIFE CYCLE, Brandon/System Press, 1970.

5. Davis, G. B., MANAGEMENT INFORMATION SYSTEMS: CONCEPTUAL FOUNDATIONS STRUCTURE, AND DEVELOPMENT, McGraw Hill, 1974.

6. Burch, John G., and Strater, Felix R., INFORMATION SYSTEMS: THEORY AND PRACTICE, Hamilton Publishing Company, 1974.

7. Milunovich, Laverne G., MANAGEMENT INFORMATION SYSTEMS, Management Services, March-April, 1970.

8. Schwartz, M. H., MIS PLANNING, Datamation, September 1, 1970.

9. Van Dusseldorp, Ralph, SOME PRINCIPLES FOR THE DEVELOPMENT OF MANAGEMENT INFORMATION SYSTEMS, Management Information Systems in Higher Education: The State of the Art, Ed. Charles B. Johnson and William G. Katzenmeyer, Duke University Press, 1969.

10. Cicio, John David., DEVELOPMENT OF A COMPUTER-BASED MANAGEMENT INFORMATION SYSTEM, Master Thesis, Naval Postgraduate School, 1972.

11. Matthews, D. Q., THE DESIGN OF THE MANAGEMENT INFORMATION SYSTEM, Auerbach Publishers, 1971.

12. Head, Robert V., MANAGER'S GUIDE TO MANAGEMENT INFORMATION SYSTEMS, Prentice Hall, 1972.

13. Kroenke, David, DATABASE PROCESSING, Science Research Associates Inc., 1977.

14. Martin, James., COMPUTER DATABASE ORGANIZATION, Prentice Hall, 1975.

15. Krauss, Leonard I., COMPUTER-BASED MANAGEMENT INFORMATION SYSTEMS, American Management Association Inc., 1970.

16. Auerbach Publishers Inc., EDP STEERING COMMITTEES, Portfolio 1-04-06, Information Management Series, 1976.

17. Wheelen, Thomas L., THE MANAGER'S AND THE SYSTEMS ANALYST'S ROLES IN MIS DEVELOPMENT, MIS in Action, Ed. Robert G. Murdick and Joel E. Ross, West Publishing Co., 1975.

18. Schneidewind, Norm F., SYSTEM ANALYSIS, DESIGN AND IMPLEMENTATION, Course Outline, Naval Postgraduate School, 1977.

19. Bingham, J. E. and Davies, G. W., A HANDBOOK OF SYSTEMS ANALYSIS, The MacMillan Press Ltd., 1972.

20. Krauss, Leonard J., ADMINISTERING AND CONTROLLING THE COMPANY DATA PROCESSING FUNCTION, Prentice Hall Inc., 1969.

21. Hartman, W., Matthes, H., and Proeme, A., MANAGEMENT INFORMATION SYSTEMS HANDBOOK, McGraw Hill Book Company, 1968.

22. Lazaro, Victor, SYSTEMS AND PROCEDURES, 2d. ed., Prentice Hall, 1968.

23. Auerbach Publishers Inc., CONFIGURING THE COMPUTER SYSTEM, Portfolio 2-04-04 (rev. 10/75), Information Management Series, 1976.

24. Auerbach Publishers Inc., SYSTEM SPECIFICATIONS, Portfolio 4-01-03, Information Management Series, 1976.

25. Auerbach Publishers Inc., SHOULD YOU HAVE A TECHNICAL WRITING STAFF, Portfolio 1-05-15, Information Management Series, 1977.

26. Shaw, John C. and Atkins, William, MANAGING COMPUTER SYSTEM PROJECTS, McGraw Hill Book Company, 1970.

27. Rubin, Martin L., DATA PROCESSING ADMINISTRATION, Handbook of Data Processing Management, Vol. 6, Auerbach Publishers Inc., 1971.

28. Kearney, James M. and Mitutinovich, Jugoslaw S., A GUIDE TO SUCCESSFUL COMPUTER SYSTEM SELECTION, Management Reference Series, Data Processing Management Association, 1976.

29. Aron, Joel D., THE PROGRAM DEVELOPMENT PROCESS: THE INDIVIDUAL PROGRAMMER, Addison Wesley, 1974.

30. Weinberg, G. M., THE PSYCHOLOGY OF COMPUTER PROGRAMMING, Van Nostrand Reinhold, 1971.

31. Yourdon, Edward, TECHNIQUES OF PROGRAM STRUCTURE AND DESIGN, Prentice Hall, 1975.

32. Van Tassel, Dennie, PROGRAM STYLE, DESIGN, EFFICIENCY, DEBUGGING, AND TESTING, Prentice Hall Inc., 1974.

33. Auerbach Publishers Inc., MANAGING THE DEVELOPMENT OF A SYSTEM, portfolio 3-03-01, Information Management Series, 1973.

34. Auerbach Publishers Inc., SYSTEM REQUIREMENTS DOCUMENTATION, portfolio 4-01-01, Information Management Series, 1975.

35. Auerbach Publishers Inc., DOCUMENTATION - MANAGEMENT PROBLEMS AND SOLUTIONS, portfolio 4-02-01, Information Management Series, 1977.

36. IBM Corporation, HIPO - A Design Aid and Documentation Technique, 2d. ed., 1975.

37. Myers, Glendford J., SOFTWARE RELIABILITY PRINCIPLES AND PRACTICES, John Wiley and Sons, 1976.

38. Huang, J. C., AN APPROACH TO PROGRAM TESTING, ACM Computing Surveys, vol. 7, number 3, September 1975.

39. Schneidewind, N. F., THE USE OF SIMULATION IN THE EVALUATION OF SOFTWARE, Computer, IEEE Computer Society, April 1977.

40. Meister, David and Rabideau, Gerald F., HUMAN FACTORS EVALUATION IN SYSTEM DEVELOPMENT, John Wiley and Sons, 1965.

41. Brooks, Frederick P., Jr., THE MYTHICAL MAN - MONTH, Addisson Wesley Publishing Company, 1975.

42. Youssef, Leon, SYSTEMS ANALYSIS AND DESIGN, Reston Publishing Company Inc., 1975.

43. Auerbach Publishers Inc., ESTABLISHING A FRAMEWORK FOR DATA BASE PLANNING, portfolio 21-02-02, Information Management Series, 1976.

44. Auerbach Publishers Inc., DATA DICTIONARY/DIRECTORY SYSTEM: A TOOL FOR DATA ADMINISTRATION AND CONTROL, portfolio 22-01-02, Information Management Series, 1977.

45. Auerbach Publishers Inc., DATA BASE MANAGEMENT SYSTEMS PERFORMANCE EVALUATION: ALTERNATIVE APPROACHES, portfolio 22-02-03, Information Management Series, 1976.

46. Auerbach Publishers Inc., DATA BASE DESIGN METHODOLOGY - PART I, portfolio 23-01-01, Information Management Series, 1976.

47. Auerbach Publishers Inc., SYSTEM DEVELOPMENT LIFE CYCLE FOR DATA BASE DEVELOPMENT, portfolio 23-02-01, Information Management Series, 1976.

48. Auerbach Publishers Inc., FUNCTIONS OF DATA BASE ADMINISTRATION (DBA), portfolio 22-05-01, Information Management Series, 1976.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93940	1
4.	Professor Norman F. Schneidewind, Code 52 Department of Computer Science Naval Postgraduate School Monterey, California 93940	1
5.	Fundacion "Gran Mariscal De Ayacucho" 515 Madison Avenue - Suite 1020 New York, NY 10022	1
6.	LCDR Gabriel Flores Prado Apartado De Correos 76850 Zona Postal 107 - El Marques Venezuela	1

Thesis
F5228
c.1

173909

Flores-Prado
A study of the system
development process.